VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Vzdělávací geoinformační aplikace pro Android

Educational Geolocation Based Application for Android

2016 Šimon Dorociak

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání diplomové práce

Student: **Bc. Šimon Dorociak**

Studijní program: N2647 Informační a komunikační technologie

Studijní obor: 2612T025 Informatika a výpočetní technika

Téma: Vzdělávací geoinformační aplikace pro Android
Educational Geolocation Based Application for Android

Jazyk vypracování: čeština

Zásady pro vypracování:

Navrhněte a implementujte naučnou geografickou aplikaci pro Android. Aplikace bude využívat lokalizační a mapové služby systému společně s volně dostupnými online informačními zdroji (Wikipedia, ELDIS, CIA World Factbook, aj.) Účelem aplikace bude zábavnou formou poukázat na zajímavá místa v okolí uživatele. Na základě zvoleného nastavení a aktuální polohy aplikace sama vybere z dostupných databází zajímavá místa a k nim vytvoří aktivity, které musí uživatel plnit (dojít na dané místo, odpovědět na kvizovou otázku, vybrat z několika možností apod.) Aktivity se budou vztahovat k místu, historickým událostem, aktuálnímu dění. Téma práce je inspirováno aktuálním úkolem z Google Summer of Code 2014.

Řešení bude obsahovat:
1. Implementaci základní mapové aplikace, volbu mapových zdrojů (Google Maps, OpenStreetMap aj.).
2. Přístup k dostupným online zdrojům, georeferencování informací.
3. Návrh a implementaci vlastního algoritmu pro výběr vhodných míst zájmů (POI). Zaměřte se na algoritmy pro automatické nalezení nejzajímavějších míst. Inspirovat se můžete například doporučenými články [4,5].
4. Dynamickou tvorbu aktivit a otázek spojených s daným POI.
5. Možnost offline uložení dat bez nutnosti stálého připojení k Internetu.
6. Testování aplikace s větším počtem uživatelů, vyhodnocení zpětné vazby.

Seznam doporučené odborné literatury:

[1] Reto Meier, Professional Android 4 Application Development, Wrox, 2012, ISBN-13: 978-1118102275
[2] Joseph Annuzzi Jr. et al., Advanced Android Application Development, Addison-Wesley Professional, 2014. ISBN 0133892387
[3] Evangelos Petroutsos, Google Maps: Power Tools for Maximizing the API, McGraw-Hill Osborne Media, 2014. ISBN 0071823026
[4] Nothegger, C., Winter, S., Raubal, M. Selection of salient features for route directions. Spatial cognition and computation, 4(2), 113-136. 2004
[5] Maervoet, J., Brackman, P., Verbeeck, K., De Causmaecker, P., & Berghe, G. V. Tour suggestion for outdoor activities. In Web and Wireless Geographical Information Systems, 54-63. 2013

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **Mgr. Ing. Michal Krumnikl, Ph.D.**

Datum zadání: 01.09.2015

Datum odevzdání: 29.04.2016

doc. Dr. Ing. Eduard Sojka
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 1. júla 2016 .....

Souhlasím se zveřejněním této diplomové práce dle požadavků čl. 26, odst. 9 Studijního a zkušebního řádu pro studium v magisterských programech VŠB-TU Ostrava.

V Ostravě 1. júla 2016 .....

Rád bych som na tomto mieste poďakoval Mgr. Ing. Michalovi Krumniklovi, Ph.D. za odborné konzultácie.

Abstrakt

Cieľom práce bolo navrhnúť a naimplementovať geovzdelávaciu aplikáciu pre operačný systém Android s použitím lokalizačných a mapových služieb samotného systému a voľne dostupných online informačných zdrojov. Účelom aplikácie je poukázať na zaujímavé miesta v najbližšom okolí užívateľa a jednoduchou a rýchlou formou daného užívateľa niečo o týchto miestach naučiť sadou dynamicky generovaných otázok vzťahujúcich sa k daným miestam, ktoré sú najdôležitejšiou častou aplikácie alebo splnením nejakej úlohy. Jednotlivé body záujmu sú dostupné taktiež offline a uložené v lokálnej databáze SQLite.

Klíčová slova: Android, bod záujmu, mapové služby, informačné zdroje, SQLite

Abstract

The goal of this thesis has been a design and implementation of geolearning application for operating system Android with an usage of localization and map services of system itself and free available online informative sources. Purpose of the application is to point out on points of interest around the user and in quick and simple way to teach user something about those places with set of dynamically generated questions which are the most important part of the application and related to those places or with fullfill of some task. Points of interest are available offline and stored in local SQLite database.

Key Words: Android, point of interest, maps services, informative sources, SQLite

Obsah

Seznam použitých zkratek a symbolů

JSON	–	JavaScript Object Notation
UML	–	Unified Modeling Language
API	–	Application Programming Interface
SQL	–	Structured Query Language
NLP	–	Natural Language Processing
HTTP	–	Hypertext Transfer Protocol
XML	–	Extensible Markup Language
POJO	–	Plain Old Java Object
UI	–	User Interface
POI	–	Point of Interest
NLTK	–	Natural Language Toolkit

Seznam obrázků

Seznam tabulek

Seznam výpisů zdrojového kódu

1 Úvod

Cieľom diplomovej práce je vytvoriť aplikáciu pre operačný systém Android, ktorá jednoduchým spôsobom vzdeláva užívateľa o rôznych zaujímavých miestach v jeho blízkom okolí za použitia lokalizačných služieb samotného systému a voľne online dostupných informácií.
Kľučová časť aplikácie je výber vhodných bodov záujmu v okolí užívateľa. Tu bolo nutné zvoliť vhodný algoritmus výberu jednotlivých bodov záujmu tak aby bol použiteľný na zdroj dát, ktorý aplikácia používa a aby bol dostatočne rýchly a užívateľ nemusel čakať dlhú dobu na výsledok. Kedže zdroje dát sú dostupné len online rovnako doležitá je aj konektivita zariadenia, ktorá je nutná k správnemu nastaveniu a inicializácii aplikácie.
Online získané data sa cachujú do lokálnej databázy takže sú tieto dáta dostupné aj v prípade, že zariadenie nemá požadovanú konektivitu.

Aplikácia po prvom spustení oboznámi užívateľa načo slúži. Následne ponúkne užívateľovi možnosť nadstaviť si profil kde užívateľ vyplní základné informácie o sebe a aplikácia následne presunie užívateľa na samotnú mapu kde po získaní aktuálnej polohy zariadenia zanalyzuje a zobrazí užívateľovi najzaujímavejšie miesta v jeho okolí. Užívateľ si môže nadstaviť do akej maximálnej vzdialenosti má aplikácia vyhľadávať a získavať tie najzaujímavejšie miesta.
Užívateľ sa môže medzi jednotlivými miestami preklikávať a získavať informácie o daných miestach formou dynamicky generovaných vedomostných otázok (ktoré sú to hlavné v rámci aplikácie), zobrazením webovej stránky miesta alebo splnením questu vzťahujúcemu sa k danému miestu, ktorý funguje tak, že užívateľ musí na dané miesto doraziť. V okamihu dorazenia je quest splnený, uložený a užívateľ ho môže zdieľať so svojimi priateľmi na Facebooku.

Motiváciou tejto práce bola skutočnosť, že som síce našiel podobné aplikácie, ktoré dokážu zistiť nejaké informácie o rôznych zaujímavých miestach (ku ktorým sa vyjadrím v sekcii podobné projekty) ale nedokázali nejakou zábavnejšou formou vzdelávania sprostredkovať užívateľovi informácie o daných miestach zaujímavých pre užívateľa. Z tohto dôvodu som sa rozhodol navrhnúť túto aplikáciu, ktorá síce ponúka podobnú funkčnosť ale dokáže navyše generovať rôzne vedomostné otázky prostredníctvom ktorých by mala užívateľa motivovať aby aplikáciu používal častejšie a istým spôsobom si ju obľúbil.

V následujúcom texte sa budem zaoberať samotným návrhom, implementáciou a realizáciou aplikácie. Obzvlášť bol kladený dôraz nato aby bola implementácia aplikácie čo najlepšia a jej popis čo najpodrobnejší. Na jeho konci sa budem zaoberať testami, vyhodnotením funkčnosti aplikácie a výhľadmi do budúcnosti.

2 Podobné projekty

Na začiatok by som rád spomenul, že hlavnou motiváciou tvorby tejto aplikácie bola skutočnosť, že som nanašiel aplikáciu, ktorá by zábavnejšou formou poskytovala užívateľovi možnosť získať nejaké vedomosti o rôznych zaujímavých miestach v jeho blízkom okolí v ktorom sa práve nachádza.
Rovnako som nenašiel aplikáciu, ktorá by dokázala podať aspoň základné informácie o nejakom bode záujmu v prípade, že je užívateľ offline, zdieľať nejaké zážitky spojené s danou aplikáciou so svojimi priateľmi.
Z týchto dôvodov som sa rozhodol pre vlastnú implementáciu takejto mobilnej aplikácie, ktorá by zábavnejšou formou dokázala užívateľa obohatiť o nové poznatky o zaujímavých miestach v jeho blízkom okolí, ktorá by dokázala s istými obmedzeniami fungovať offline, poskytovala možnosť zdieľať zážitky s priateľmi a spĺňala najnovšie trendy užívateľského rozhrania.

Teraz by som podal nejaké informácie o podobných projektoch a dané projekty porovnal.

2.1 Wikipedia App

Túto aplikáciu som našiel ako prvú. Je to aplikácia priamo od *wikipédie*. Prirovnal by som ju priamo k mobilnej verzii webovej stránky wikipédie. Inými slovami neponúka nič extra navyše. Mám na mysli funkcionalitu, ktorá by zábavnejšou formou dokázala užívateľovi sprostredkovať nové vedomosti o rôznych zaujímavých miestach v jeho okolí bez toho aby ich musel sám užívateľ pracne vyhľadávať. Na druhej strane konkrétne táto aplikácia je navrhnutá v material dizajne takže užívateľské rozhranie má veľmi pekné viz. obrázky 1 a 2.

Obrázek 1: Ukážka UI Wikipedia App 1

Obrázek 2: Ukážka UI Wikipedia App 2

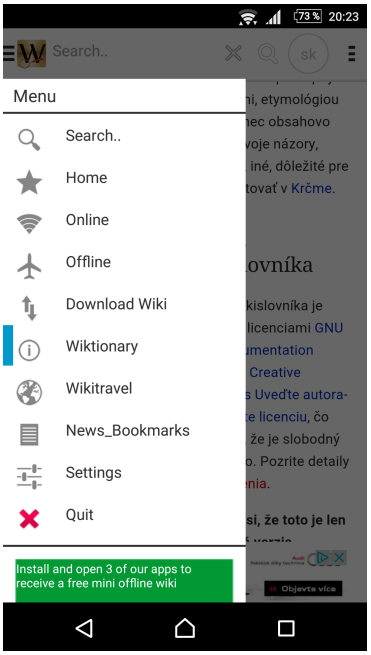
Teraz by som zhrnul v jednotlivých bodoch jej výhody a nevýhody:

- Užívateľské rozhranie je v material dizajne
- Možnosť zobrazenia náhodného článku
- História vyhľadávania
- Priamo podpora od wikipédie
- Chýba nejaká zábavnejšia forma vzdelávania (vedomostné otázky napríklad)
- Questy spojené s nejakým zaujímavým miestom

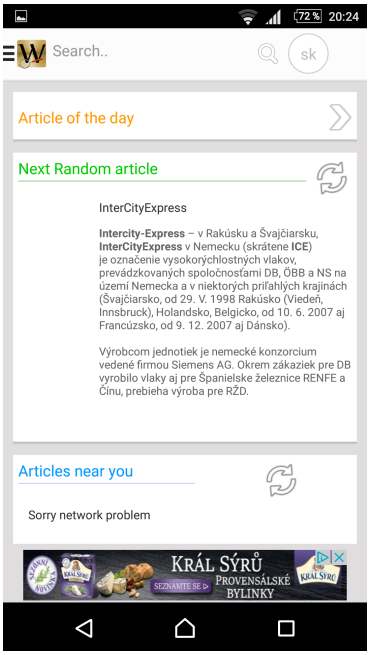
Aplikáciu by som opísal tak, že je to kvalitná aplikácia s mnohými funkciami ale chýba tam už spomenutá zábavnejšia forma vzdelávania sadou nejakých otázok alebo questov, ktoré by užívateľa donútili napríklad osobne doraziť na konkrétne miesto a tam sa priamo zapojiť do vzdelávania ci už zakúpením vstupenky alebo sprievodcu.

2.2 Wiki Encyclopedia: Wikipedia

Ďalšia podobná aplikácia, ktorú som našiel s využitím voľne dostupných dát z wikipédie. Viacmenej platí pre túto aplikáciu to isté čo pre aplikáciu v predchádzajúcom texte. Aplikácie sú veľmi podobné čo sa týka funkcionality.
Táto aplikácia má však navyše možnosť si stiahnúť do mobilu istú verziu databázy wikipédie aby boli informácie prístupné offline ale táto možnosť je spoplatnená takže normálne nie je k dispozícii.
Celkovo je však aplikácia málo funkčná a nejaké veci tam nefungujú (môže to byť spôsobené samotnou aplikáciou alebo zmenami serveru odkiaľ aplikácia získava údaje na ktoré aplikácia nereagovala). Rovnako ako aplikácia od wikipédie ani táto aplikácia neobsahuje funkcionalitu zábavného podania vedomostných informácií.
Ako hlavné mínus vidím to, že užívateľské rozhranie nie je v material dizajne. Z tohoto dôvodu aplikácia nepôsobí (aspoň na mňa) moc pozitívne a nevidím žiadny dôvod aby som túto aplikáciu používal. Ukážka užívateľského rozhrania je na obrázkoch 3 a 4.



Obrázek 3: Ukážka UI Wiki Encyclopedia 1



Obrázek 4: Ukážka UI Wiki Encyclopedia 2

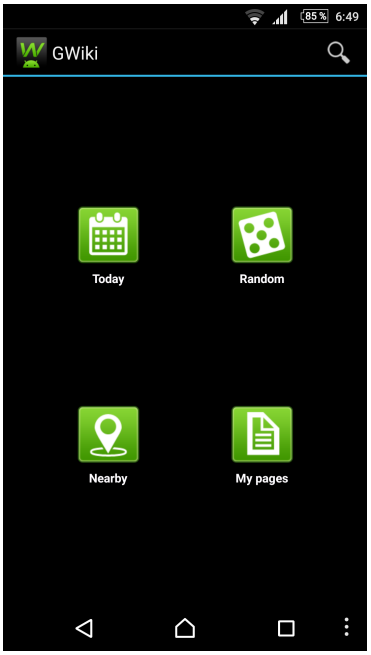
Ako je vidieť na obrázkoch tak daná aplikácia používa ešte stále staré postupy pre návrh užívatelského rozhrania, ktoré sú matúce a nepôsobia moc pekne (farby, písmo, ikonky, obsah).

2.3 GWiki

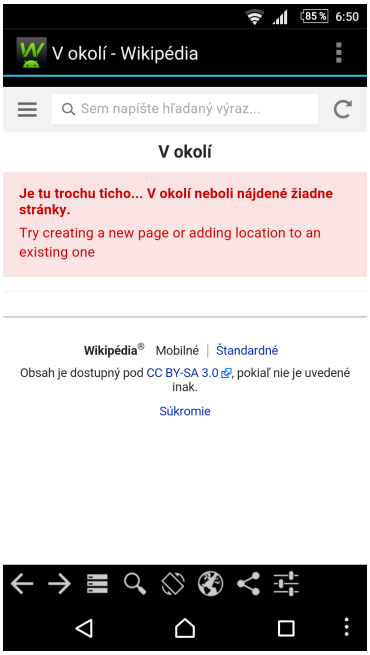
GWiki je ďalšia mobilná aplikácia, ktorá dokáže užívateľovi sprostredkovať informácie získané z wikipédie. Dá sa považovať asi za najjednoduchšiu z aplikácií, ktoré som v tejto sekcii uviedol a ešte uvediem. Obsahuje následujúce funkcie:

- Zobrazenie 'dnešného' článku z wikipédie
- Zobrazenie náhodného článku
- Zobrazenie nejakých článkov v okolí (daná funkcionalita ale nefunguje)
- Zobrazenie mnou uložených stránok (bookmarks)

Nefunguje však v offline režime a podľa všetkého asi necachuje obsah. Podobne ako mobilná aplikácia *Wiki Encyclopedia: Wikipedia* nemá užívateľské rozhranie navrhnuté v material dizajne viz. obrázky 5, 6 ale z môjho pohľadu je to UI lepšie ako v spomenutej aplikácii.



Obrázek 5: Ukážka UI GWiki 1



Obrázek 6: Ukážka UI GWiki 2

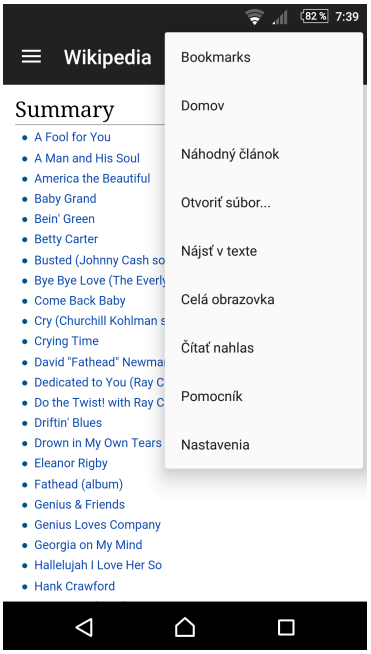
Ako je vidiet na vyššie uvedených ukážkach užívateľského rozhrania tak aplikácia má na svojej 'home' obrazovke použité 'grid' menu, ktoré nie je síce v material dizajne ale je jednoduché a 'user-friendly' takže užívateľ vie presne čo to menu robí. V ďalšej ukážke môžeme vidieť spodné menu, ktoré má zlé zarovnanie ikoniek (celkovo ikonky sú na displejoch z vyšším rozlíšením rozmazané) a chýba im 'touchback feedback' čo znamená, že keď užívateľ na danej ikonke v menu podrží prst dlhšiu dobu, menu by mu malo zobraziť výstižný popis funkcie, ktorou ovplýva čo sa tu nedeje a pre užívateľa to môže pôsobiť zmätene. Ale celkovo užívateľské rozhranie tu je z hľadiska prehľadnosti lepšie ako v aplikácii *Wiki Encyclopedia: Wikipedia*.

2.4 Kiwix

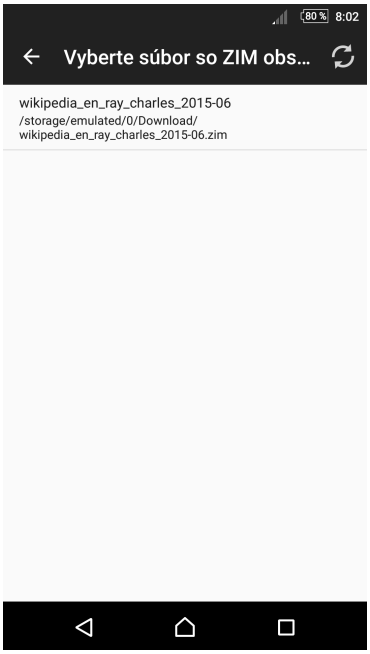
Je poslednou aplikáciou, ktorú spomeniem a v jednoduchosti popíšem v tejto sekcii. Konkrétne táto aplikácia ma zaujala zo všetkých najviac. Je priamo navrhnutá pre offline prístup k dátam k wikipédii za použitia material dizajnu viz. obrázky 7, 8. Inými slovami funguje doslova ako webová stránka ale čisto offline. Funguje na princípe sťahovania **.zim** súborov, ktoré obsahujú dáta z wikipédie ku konkrétnemu článku, osobe, skupine článkov (záleží aký súbor si užívateľ stiahne podľa dostupnosti). Užívateľ následne v aplikácii nadstaví cestu k danému súboru a aplikácia zobrazí informácie z daného súboru v zozname. Užívateľ si môžem následne prezerať jednotlivé články bez obmedzenia pretože sú lokálne uložené v zariadení.

Výhoda tejto aplikácie spočíva v tom, že je kompletne navrhnutá len pre offline prístup takže nie je závislá na žiadnom online zdroji dát. Ďalej musím vydvihnúť funkciu čitania článku čo je veľmi užitočná funkcia napríklad keď cestujete a nechce sa Vám čítať. Môžeme to taktiež považovať za istú formu výuky anglickej výslovnosti počas počuvania 'native speakera'. Celkovú funkcionalitu mobilnej aplikácie a jej nedostatky zhrniem v následujúcich bodoch:

- Čítanie textu (anglicky)
- Nočný režim (biela farba pozadia sa zmení na čiernu a text na tmavo žltú a bielu farbu)
- Kompletný prístup k offline wikipédii
- Úplná kontrola nad zdrojom dát vo forme cachovaných súborov na uložisku
- Jednoduchosť
- Užívateľské rozhranie za použitia Material dizajnu
- Zobrazenie náhodného článku
- Neobsahuje funkciu, ktorá by zábavnejším spôsobom poskytla užívateľovi zaujímavé informácie



Obrázek 7: Ukážka UI Kiwix 1



Obrázek 8: Ukážka UI Kiwix 2

V uvedených ukážkach môžeme vidieť bohaté menu aplikácie ktoré obsahuje záložky, ktoré obsahujú články, ktoré si užívateľ pridal do svojich obľubených, možnosť otvorenia nové súboru

(.zim) s cieľom načítať ďalšie články, spomenutú funkciu čitania textu alebo vyhľadávanie textu v článkoch.
V druhej ukážke je vidieť obrazovku v ktorej sa nachádzajú **.zim** súbory, ktoré si užívateľ stiahol [1] do mobilného zariadenia. K tomuto nemám viacmenej čo vytknúť ale bolo by fajn keby sa dali dané súbory mazať a užívateľ by nemusel tie súbory mazať cez nejakého manažéra súborov.

2.5 Zhrnutie

Z aplikácií uvedených v tejto sekcii ma najviac zaujala tá posledná teda **Kiwix** pretože poskytuje úplný prístup k článkom wikipédie offline čo je veľmi výhodné keď cestuje človek napríklad v metre kde nie je väčšinou signál, podporuje anglické čítanie textu, má vytvorené oficiálne webové stránky odkiaľ si môže užívateľ sťahovať dáta, ktoré potrebuje a zistiť nejaké ďalšie informácie týkajúce sa samotnej aplikácie. Navyše je užívateľské rozhranie navrhnuté v material dizajne čo je veľké plus pretože drží krok s dobou narozdiel od niektorých iných aplikácií. Určite aplikáciu ešte niekedy použijem.

Na trhu existujú aplikácie, ktoré poskytujú podobnú funkcionalitu akou je napríklad podávanie voľne dostupných informácií o zaujímavých miestach z wikipédie, poskytujú mapu, náhodné generovanie 'článku dňa', úplný prístup offline bez obmedzení a ďalšie podobné funkcie avšak žiadna neponúka (aspoň som takú nenašiel) funkcionalitu, ktorá by priamo nejakou zábavnejšou formou poskytla užívateľovi informácie o rôznych zaujímavých miestach v jeho blízkom okolí bez pracného ručného vyhľadávania alebo umožnila plniť užívateľovi nejaké úlohy (questy) spojené s nejakým konkrétnym bodom záujmu a dokončené úlohy zdieľať so svojimi priateľmi formou nejakého vygenerovaného obrázku napríklad.

Táto skutočnosť ma utvrdila v tom, že by som mal vytvoriť mobilnú aplikáciu, ktorá by podobnú funkcionalitu umožňovala.

[1] Dostupné na: $http://www.kiwix.org/wiki/Main_Page$

3 Analýza riešenia

V predchádzajúcom texte boli obsiahnuté základné informácie o aplikácii a jej funkčnosti. Načrtol som motiváciu tvorby tejto aplikácie a spomenul podobné projekty, ktoré fungujú na podobnom princípe aj keď neponúkajú funkcie, ktoré ponúka táto mobilná aplikácia. Táto kapitola sa bude venovať kompletnej analýze požiadavkov na základe ktorej bude prebiehať implementácia. Tu zhrniem a popíšem funkcie, ktorými bude aplikácia disponovať a načrtnem spôsob ich implementácie.

3.1 Získavanie bodov záujmu

Dôležitou časťou mobilnej aplikácie je spôsob získavania bodov záujmu v okolí užívateľa. Bolo nutné ho navrhnúť tak, aby bol použiteľný pre mobilnú aplikáciu, ktorá má odlišný životný cyklus [1, 2] od aplikácie napríklad pre desktop a zároveň nebol veľmi pomalý a náročný na výkon.

Ako zdroj dát z ktorých aplikácia získavá body záujmu je použitá webová služba Google Places API priamo od spoločnosti Google, ktorá je pre účel aplikácie vhodná. Umožňuje získavať potrebné dáta bodov záujmu dotazovaním sa na danú webovú službu prostredníctvom HTTP requestov prostredníctvom ktorých je možné presnejšie definovať 'čo chceme aby nám služba vrátila'. Napríklad môžeme špecifikovať konkrétnu kategóriu miesta záujmu (napr. church) alebo dôležitejšie môžeme definovať maximálny rádius okolia mobilného zariadenia v ktorom bude služba vyhľadávať [7]. Z vyššie uvedených dôvodov som nepoužil integráciu *OpenStreetMap*, ktoré mi nevyhovovali z hľadiska integrácie s Androidom (napríklad samotné mapové súbory dosahujú obrovských veľkostí).

Bližšie informácie poskytnem v sekcii v ktorej budem popisovať implementáciu aplikácie.

3.1.1 Bod záujmu

Bod záujmu môžeme opísať ako konkrétne miesto na Zemi, ktoré je definované pomocou súradníc tzv. zemepisnej dĺžky a šírky, názvom, typom (kategóriou) a ďalšími doplnkovými informáciami, ktoré sa špecificky líšia od konkrétneho typu. Dôležité je podoknúť, že toto miesto je istým spôsobom významné pre človeka ale aj pre samotnú Zem. Môže sem patriť napríklad kostol, múzeum, univerzita a podobne.

3.1.2 Dôležitosť bodu záujmu

Dôležitosť bodu záujmu môžeme vyjadriť množstvom informácií, ktoré sa vzťahujú k danému bodu. Čím viac informácií a dát sa vzťahuje ku konkrétnemu bodu záujmu, tým viac môžeme považovať daný bod záujmu (point of interest, skr. POI) za dôležitejší respektíve 'zaujímavejší'.

Tento algoritmus môžeme prirovnať k algoritmu Katz Centrality, ktorý vyjadruje dôležitosť uzla v sieti. Čím viac má daný uzol priamych susedov (first degree nodes) tým má väčšiu dôleži-

tosť v rámci siete [4]. Pracujú rozdielne ale princíp je rovnaký. V mobilnej aplikácii nás zaujímajú práve tie body záujmu, ktoré sú najdôležitejšie.

Rád by som ešte na tomto miesto napísal, že som sa teda rozhodol v aplikácii použiť vlastný algoritmus získavania POI, ktorý som popísal v texte vyššie. Ale taktiež veľmi zaujímavé algoritmy sú popísané v článkoch *Selection of Salient Features for Route Directions* a *Tour Suggestion for Outdoor Activities*, ktoré som naštudoval a z časti sa nimi inšpiroval. Ponúkajú veľmi zaujímavé pohľady a prístupy k výberu bodov záujmu ale v rámci tejto verzie mobilnej aplikácie nie sú použité pretože som použil vyššie spomenutý vlastný algoritmus.

3.2 Dynamické generovanie otázok k bodom záujmu

Ďalšou dôležitou časťou mobilnej aplikácie bolo navrhnúť algoritmus pre dynamické generovanie otázok k jednotlivým bodom záujmu bez nutnosti vlastnej databázy otázok aby to bolo skutočne dynamické. Princíp mal byť taký, že algoritmus na základe špecifickej frázy (v našom prípade názvu bodu záujmu) vygeneruje sadu otázok v kontexte danej frázy.
To znamená, že k fráze 'Britské múzeum' algoritmus vygeneruje otázky, ktoré sa vzťahujú k danému múzeu.
K dosiahnutiu tohto cieľa algoritmus používa Natural Language Processing (NLP) a dáta z wikipédie.

3.3 Dátový formát pre offline prístup

Ďalším požiadavkom na aplikáciu bola možnosť prístupu k dátam offline. Kedže mobilná aplikácia pracuje s dátami výhradne dostupnými online je táto funkcionalita užitočná v prípade, že užívateľ nemá prístup k internetu a chce splniť nejaký quest v aplikácii, ktorý je možné splniť aj v offline režime. Následne bolo nutné zvoliť vhodný formát pre persistenciu dát (v našom prípade bodov záujmu), ktoré budú dostupné lokálne v mobilnom zariadení. Zdroj dát (webová služba Google Places) vracia data štruktúrovane vo forme *XML* alebo *JSON* [7].

Pre mobilnú aplikáciu bol použitý formát *JSON* z následujúcich dôvodov [8][9]:

- je 'lightweight' formát pre výmenu dát
- je jednoduchý na čítanie a na samotné vytváranie pre ľudí a samotné počítače (mašiny)
- kompletne nezávislý na použitom programovacom jazyku
- je to dátovo orientovaný formát
- ľahšie mapovateľný na objektovo-orientované systémy

Následne sú dáta vrátené v *JSON* formáte sparsované prostredníctvom POJO tried a finálne uložené do SQLite databázy, ktorá je použitá ako konečný dátový formát dát (body záujmu) pre

offline prístup. Veci spojené s databázou ako je databázová schéma, štruktúra, dotazy a prístup budú vysvetlené v sekcii implementácia.

3.3.1 Ukážka formátu zdroja dát

Následujúce výpisy 1 a 2 predstavujú ukážku zdrojových dát získaných z webovej služby Google Places, ktoré mobilná aplikácia získava a parsuje. K jednotlivým výpisom je pridaný popis atributov, ktoré aktívne využíva mobilná aplikácia.

```
{
   "results" : [
      {
         "geometry" : {
            "location" : {
               "lat" : -33.870775,
               "lng" : 151.199025
            }
         },
         "icon" : "http://maps.gstatic.com/mapfiles/place_api/icons/tra...",
         "id" : "21a0b251c9b8392186142c798263e289fe45b4aa",
         "name" : "Rhythmboat Cruises",
         "opening_hours" : {
            "open_now" : true
         },
         "photos" : [
            {
               "height" : 270, "photo_reference" : "CnR...", "width" : 519
            }
         ],
         "place_id" : "ChIJyWEHuEmuEmsRm9hTkapTCrk",
         "reference" : "CoQBdQAAAFSi....",
         "types" : [ "travel_agency", "restaurant", "food", "establishment" ],
         "vicinity" : "Pyrmont Bay Wharf Darling Dr, Sydney"
      }
   ],
   "status" : "OK"
}
```

Výpis 1: Google Places JSON [7]

Vo vyššie uvedenom výpise môžeme vidieť reprezentáciu bodu záujmu vo formáte *JSON*. V rámci aplikácie sú využívané následujúce atribúty:

- objekt **location** ktorý predstavuje zemepisné súradnice bodu záujmu tzv. zemepisnú šírku a zemepisnú dĺžku. Toto je viacmenej najdôležitejší parameter popisujúci geo polohu bodu záujmu.
- parameter **name**, ktorý reprezentuje názov miesta záujmu. Tento parameter je tiež veľmi dôležitý pretože predstavuje samotný názov miesta a zároveň na základne tohto parametru aplikácia dynamicky generuje sadu vedomostných otázok ku ktorým sa vyjadrím bližšie v sekcii implementácia.
- pole **photos**, ktoré obsahuje fotky daného bodu záujmu. Tento parameter nie je povinný takže sa môže stať, že ho webová služba z ktorej aplikácia získavá data (Google Places) nemusí vrátiť. V tomto príde ho aplikácia ignoruje.
- parameter **id**, ktorý jednoznačne identifikuje daný bod záujmu. Jeho hashovaná forma je použitá lokálnou databázou ako primárny kľúč s cieľom poskytnúť body záujmu aj v offline režime.
- parameter **placeId**, ktorý sa používa pre získanie dodatočných parametrov daného bodu záujmu ako je napríklad webová stránka daného miesta alebo ďalšie obrázky.
- parameter **photoReference** predstavuje konkrétnu referenciu na fotku daného miesta pomocou ktorej sa z webovej služby získajú data konkrétneho obrázku.

```
{
   "html_attributions" : [],
   "result" : {
      "address_components" : [ ... ],
      "formatted_address" : "48 Pirrama Road, Pyrmont NSW, Australia",
      "formatted_phone_number" : "(02) 9374 4000",
      "geometry" : { ... },
      "icon" : "http://maps.gstatic.com/mapfiles/place_api/icons/
         generic_business-71.png",
      "id" : "4f89212bf76dde31f092cfc14d7506555d85b5c7",
      "international_phone_number" : "+61 2 9374 4000",
      "name" : "Google Sydney",
      "place_id" : "ChIJN1t_tDeuEmsRUsoyG83frY4",
      "scope" : "GOOGLE",
      "alt_ids" : [ ... ],
```

```
      "rating" : 4.70,
      "reference" : "CnRsAAAA98C4wD-VFvzGq-KHVEFhlHuy1TD1W6UYZw7KjuvfVsKMRZkbCV
          ...,
      "reviews" : [
          {
          "aspects" : [... ],
           "author_name" : "Simon Bengtsson",
           "author_url" : "https://plus.google.com/104675092887960962573",
           "language" : "en",
           "rating" : 5,
           "text" : "Just went inside to have a look at Google. Amazing.",
           "time" : 1338440552869
        },
        {
          "aspects" : [... ],
           "author_name" : "Felix Rauch Valenti",
           "author_url" : "https://plus.google.com/103291556674373289857",
           "language" : "en",
           "rating" : 5,
           "text" : "Best place to work :-)",
           "time" : 1338411244325
        }
      ],
      "types" : [ "establishment" ],
      "url" : "http://maps.google.com/maps/place?cid=10281119596374313554",
      "vicinity" : "48 Pirrama Road, Pyrmont",
      "website" : "http://www.google.com.au/"
   },
   "status" : "OK"
}
```

Výpis 2: Google Places Detail JSON [7]

Vo vyššie uvedenom výpise je uvedený zdroj dát, ktorý mobilná aplikácia používa pri zobrazení detailu bodu záujmu. Následne popíšem parametre, ktoré aplikácia využíva:

- parameter **formattedAddress**, ktorý predstavuje kompletnú naformatovanú adresu bodu záujmu. Je vhodné spomenúť, že webová služba vracia rovnakú adresu avšak rozdelenú na jednotlive časti ako pole **addressComponents** čo je veľmi výhodné ale v rámci tejto aplikácie tento atribút nie je použitý pretože si ho samotná aplikácia nevyžaduje.
- parameter **internationalPhoneNumber**, ktorý reprezentuje medzinárodné telefonné číslo, ktoré je veľmi užitočné pre samotného užívateľa a môže ho využiť v prípade, že sa nachádza v blízkosti daného miesta a nevie si ďalej rady. Z tohto dôvodu je použité aj v mobilnej aplikácii.
- pole **reviews**, ktoré mobilná aplikácia používa aby bola schopná zobraziť užívateľovi recenzie o konkrétnom bode záujmu čo je taktiež veľmi dôležité pre užívateľa pretože týmto spôsobom si jednoducho vyberie, ktoré miesto navštívi a ktoré nenavštívi.
- parameter **rating**, ktorý predstavuje celkové hodnotenie bodu záujmu. Mobilná aplikácia ho využíva aby podala vizuálnu reprezentáciu hodnotenia bodu záujmu vo forme hviezdičiek.
- parameter **website** predstavujúci webovú stránku bodu záujmu. Taktiež veľmi užitočný parameter, ktorý aplikácia využíva. Užívateľ si môže prezrieť oficiálne webové stránky miesta kde si môže napríklad nájisť cenu vstupného, otváracie hodiny atď.
- pole **types**, ktorý predstavuje kategóriu miesta záujmu. Mobilná aplikácia ho spracúvava ale jeho použitie bude zaradené až v ďalšej verzii aplikácie. Bližšie info podám vo 'výhľadoch do budúcnosti'.

V následujúcom výpise 3 je ukážka *URL*, ktorú mobilná aplikácia používa pre získanie obrazových dát tzv. obrázku ku konkrétnemu bodu záujmu. Ako povinné parametre sú použité:

- **maxWidth** určujúca maximálnu šírku obrázku. V aplikácii sa za tento parameter dosádza šírka displeja. Je to veľmi dôležité z hľadiska dodržania určitej *bandwidth* aby sa nesťahoval pre 800x480 pixelový displej obrázok vo FHD rozlíšení.
- **photoreference**, ktorá jednoznačne určuje obrázok, ktorý má webová služba vrátiť.
- **apiKey** definujúci 'zariadenie', ktoré si žiada o odoslanie obrázku.

```
https://maps.googleapis.com/maps/api/place/photo?maxwidth=400&photoreference=
    CnRtAAAATLZNl354RwP_9UKbQ_5Psy40texXePv4oAlgP4qNEkdIrkyse7rPXYGd9D_Uj1rVsQdWT4oRz4QrYA
    &key={api_key}
```

Výpis 3: Google Places Photo URL [7]

Na vyššie uvedenú *URL* sa z mobilnej aplikácie pošle *HTTP* request, ktorý spracuje odpoveď a z odpovede spracuje získaný obrázok. Bližšiu špecifikáciu uvediem v sekcii implementácia.

3.4 Questy spojené s bodom záujmu

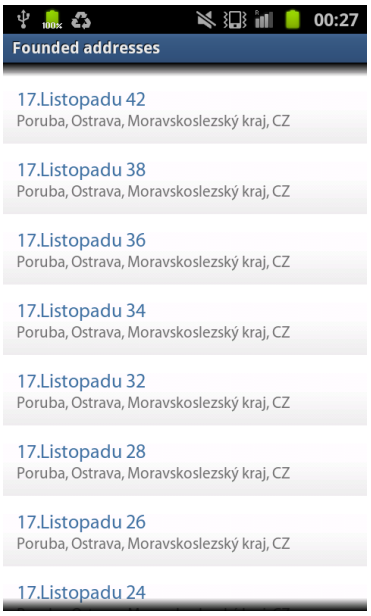
Questy predstavujú rôzne úlohy spojené s konkrétnym bodom záujmu, ktoré môže užívateľ v rámci aplikácie 'splniť'. Cieľom týchto questov (úloh) je myšlienka motivovať užívateľa k zisteniu nejakej informácie o danom bode záujmu a tým pádom sa istým spôsobom vzdelávať čo je primárna úloha mobilnej aplikácie.
Spomenuté vzdelávanie nemusí nutne prebiehať iba formou získavania informácií ale rovnako ako získavanie informácií online alebo z kníh môže vzdelávanie prebiehať aj tak, že užívateľ dané miesto fyzicky navštívi a na danom mieste sa bude vzdelávať (zaplatí si sprievodcu, prečíta si informačné tabule, atď.).
Súčasná verzia aplikácie ponúka momentálne quest 'doraziť na miesto' čo znamená, že aplikácia bude sledovať miesto a v okamihu keď sa užívateľ dostane s mobilným zariadením fyzicky na sledované miesto je quest splnený a uložený v databáze. Výsledok questu bude môcť užívateľ zdieľať na sociálnej sieti.
Ďalšie formy questov neboli z časových dôvodov zaradené do aplikácie pretože si vyžadujú kompletne 'customizované' vlastné webové rozhranie a API aby sa splnili ciele a požiadavky. Tieto questy budú spomenuté vo výhľadoch do budúcnosti.

4 Návrh realizácie

V tejto sekcii sa budem venovať návrhu užívateľského rozhrania. Obzvlášť pre mobilné zariadenie je správny návrh užívateľského rozhrania kľúčový pretože uživateľ sa musí pri používaní aplikácie citiť príjemne, aplikácia musí na užívateľa pôsobiť pozitívne. Pokiaľ mobilná aplikácia nemá 'krásne' užívateľské rozhranie tak ju užívateľ nebude používať. Kedže sa už niekoľko rokov živím ako vývojár mobilných aplikácií pre platformu Android tak môžem potvrdiť, že z osobných skúseností aj so skúsenosťami s inými firmami je správny návrh UI to najdôležitejšie.

Čo sa týka konkrétne Androidu tak v jeho prvých verziách (2,3,4) neexistoval žiadny unifikovaný spôsob ako navrhovať užívateľské rozhranie. Jednoducho sa mobilné aplikácie tvorili 'nejakým' spôsobom a výsledkom bola široká škála aplikácií, ktoré mali veľmi roznorodné užívateľské rozhranie (niekedy až bizardné). Ukážka užívateľského rozhrania so staršej verzie systému Android je na obrázkoch 9 a 10.

Obrázek 9: Ukážka UI 1 pre Android 2



Obrázek 10: Ukážka UI 2 pre Android 2

Ako je vidieť na vyššie uvedených obrázkoch tak vzhľad aplikácie na Androide 2 nepôsobí na užívateľa moc pozitívne aj keď UI v príklade je ešte v norme. Z časti je to spôsobené tým, že Android 2 obsahuje omnoho menej možností tvorby vzhľadovo pôsobivejšieho užívateľského rozhrania.

V následujúcom zozname vypíšem tie najhlavnejšie veci, ktoré chýbajú v Androide 2 v porovnaní s Android 3 a vyššie [2, 3, 14]:

- Absencia *ActionBaru*, ktorý predstavuje zjednodušene istú formu 'užívateľského panelu' spolu s možnosťou špecifikácie názvu obrazovky a pridania menu, spinnera atď.
- Moderný prístup k tvorbe obrazoviek vo forme *Fragmentov*. Dovtedy každá obrazovka predstavovala len jednoduchú 'aktivitu'.
- Veci spojené s implementáciou a grafikou napríklad *content loaders*, *resizable widgets* alebo hardvérovo zrýchlená grafika, animácie na základne 'properties' widgetov a ďalšie.

Z vyššie uvedeného zoznamu to nevyzerá, že toho chýba veľké množstvo ale skutočne je vývoj Androidu 2 značne limitovaný v porovnáni s jeho novšími verziami a jednoducho 'krásnu' aplikáciu sa na Android 2 nedá spraviť a v sučasnosti sa už ani nevyvíjajú aplikácie pre verziu Androidu staršiu ako 4 pretože by mala daná aplikácia veľké obmedzenia z hľadiska UI a funkčnosti.

Rád by som poznamenal, že existujú rôzne knižnice a API vyvinuté vývojarmi, ktoré sú publikované na *Githube*, ktoré umožňovali použiť napríklad *ActionBar* na Androide 2 (pre zaujímavosť sa daná knižnica volá ActionBarSherlock).

V súčasnej dobe (2015, 2016) sa už netvoria aplikácie s podporou Androidu 2 ale rôzne 'support' knižnice áno. V odosť väčšej intenzite od minulého roka kedy spoločnosť Google predstavila **material dizajn** ako nový, unifikovaný smer ako vytvárať užívateľské rozhranie aby bolo naozaj jednotné a aby keď sa na aplikáciu pozrie užívateľ tak si povedal, že je to skutočne aplikácia pre Android a aby naňho pôsobila pozitívne a rád ju používal.

Spomenuté 'support' knižnice sa používajú hlavne na spätnú kompatibilitu material dizajnu na Androide 4 pretože zariadení s Androidom 5 a 6 je stále len mizivé percento oproti zariadeniam s Androidom 4. Bližšie sa k material dizajnu vyjadrím v následujúcom texte.

4.1 Užívateľské rozhranie

Užívateľské rozhranie (UI) je mnohými odborníkmi na danú problematiku považované za základný pilier mobilnej aplikácie. A ja im dávam rovnako za pravdu. UI je prvá vec s ktorou príde užívateľ do styku po spustení aplikácie na mobilnom zariadení.
Prostredníctvom užívateľského rozhrania aplikácia komunikuje s užívateľom takže už len táto skutočnosť dáva naznámosť, že je nutné mu venovať dostatok času aby bolo navrhnuté:

- intuitívne bez zbytočných zložitých a nejasných prvkov
- vhodne zvolené farby a ich kombinácie
- nekombinovať veľa farieb (vybrať si 2-5 farieb charakterizujúce 'look and feel' aplikácie a tých sa držať)
- malo by pôsobiť na užívateľa pozitívne a nemalo by v ňom vzbudzovať negatívne pocity
- vhodná typografia a čitateľnosť textu
- dodržiavať 'guidelines', ktoré sú vytvárané dlhoročnými skúsenosťami (ak sú dostupné)
- rýchly prístup k cieľovým informáciam

Užívateľské rozhranie mobilnej aplikácie je navrhnuté kompletne v material dizajne, ktorému sa budem venovať v následujúcej podsekcii. Cieľom návrhu bolo navrhnúť užívateľské rozhranie tak aby spĺňalo najnovšie trendy a aby na užívateľa pôsobilo pozitívne a aby sa k aplikácii rád vracal. Celkový grafický návrh aplikácie je k dispozícii v prílohe.

4.1.1 Material dizajn

Ako som spomenul v prechádzajúcom texte tak mobilná aplikácia je kompletne navrhnutá v material dizajne s cieľom aby spĺňala najnovšie trendy a hlavne bližšie priblížiť samotný material dizajn a ukázať, že týmto krokom Google započalo cestu, ktorou by sa mal každý Android vývojár uberať pretože je to viacmenej prvý unifikovaný princíp ako tvoriť užívateľské rozhranie Android aplikácií a ako to robiť správne.
V následujúcom texte ho teda popíšem a vydzvihnem jeho hlavné vlastnosti.

Ako som už spomenul je to nový spôsob tvorby dizajnu mobilných aplikácií pre platformu Android s ktorým prišiel Google s vydaním verzie Androidu 5. Predstavuje kompletne nový prístup k tvorbe užívateľského rozhrania prostredníctvom takzvaných 'material' prvkov [10], ktoré sú špecifické a jednotné svojimi vlastnosťami, rozmery atď. Na začiatok by som spomenul prečo vlastne vznikol material dizajn:

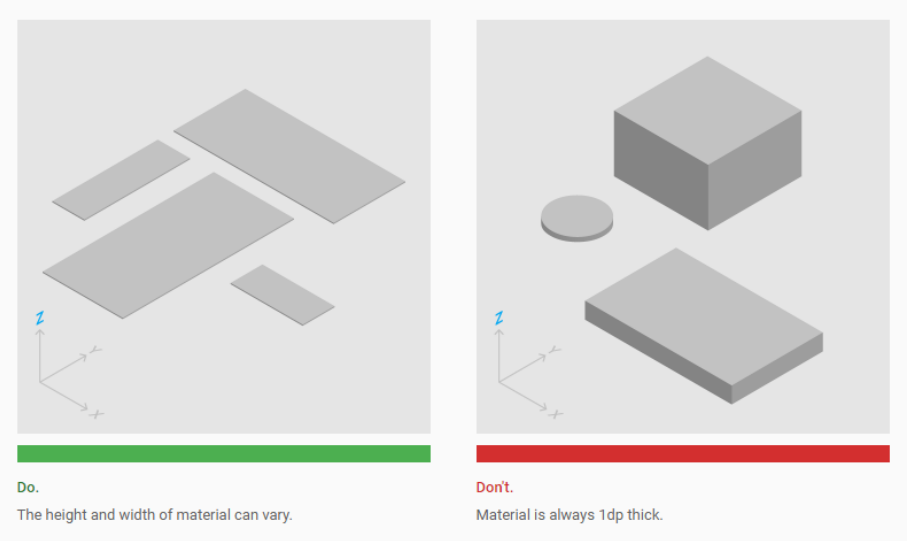
- Cieľom bolo vytvorenie vizuálneho jazyka, ktorý spája klasické princípy dobrého dizajnu s inováciou a možnosťami vedy a techniky.
- Išlo o vytvorenie jednotného systému, ktorý umožuje unifikovaný prístup naprieč platformami a zariadeniami. Benifity tohto prístupu sú hlavne vo vzhľade samotných aplikácií, ktoré sú navrhnuté odteraz jednotne (UI) takže sa nestane, že si uživateľ alebo vývojár pri pohľade na návrh užívateľského rozhrania povie, že je to aplikácia pre Apple alebo Windows Phone.

Ako som už spomenul tak material dizajn je komplexná sada pravidiel, ktoré sa musia dodržiavať. Tie najhlavnejšie pravidlá sa budem snažiť napísať v následúcom zozname:

- Prvky majú meniace sa x a y rozmery (merané v dp[2]) a rovnomernú hrúbku, ktorá je fixne stanovená na 1dp
- Tiene prvkov sú tvorené prirodzene na základne relatívnej výšky (pozícia osi z)
- Viacnásobné prvky nemôžu zaberať rovnaký bod v priestore súčasne
- Obsah je zobrazený v akomkoľvek tvare, farbe avšak nesmie meniť 'thickness' material dizajnu (1dp) viz. obrázok 11
- Obsah sa môže 'správať' nezávisle na material dizajne (tým je myslené umiestnenie, animácie, atď) avšak len v rámci hraníc material dizajnu. Hranicami sa myslia defaultné medzery, ktoré by mali mať jednotlivé material prvky (napríklad text by mal byť od okraja 16dp).
- Material dizajn je 'celistvý' tzv. napríklad touch event nemôže zasahovať do pozadia, ktoré je za prvkom, ktorý aktuálne získal touch eventu. viz obrázok 12.

[2]DP - Density-independent pixels

- Material dizajn môže meniť tvar prvkov (napríklad animácia zmeny ikony menu na ikonu navigačnej šipky)
- Material prvky sa neohýbajú a neskládajú
- Prvky môžu byť kedykoľvek živelne vytvorené a kedykoľvek môžu byť zničené (napríklad fade-in a fade-out animácia)
- Prvky sa môžu pohybovať po akejkoľvek osi (x,y,z)



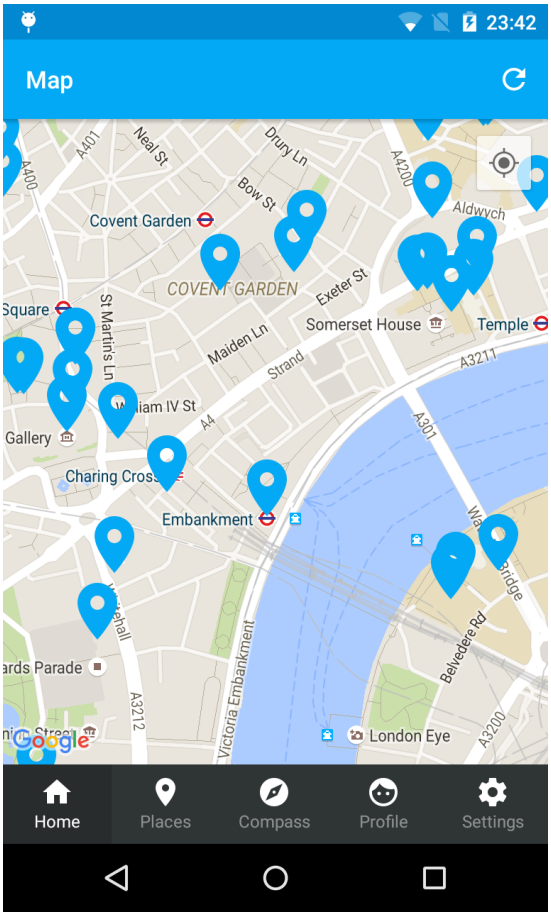
Obrázek 11: Ukážka Material Thickness Pravidla

Obrázek 12: Ukážka Material Solid Pravidla

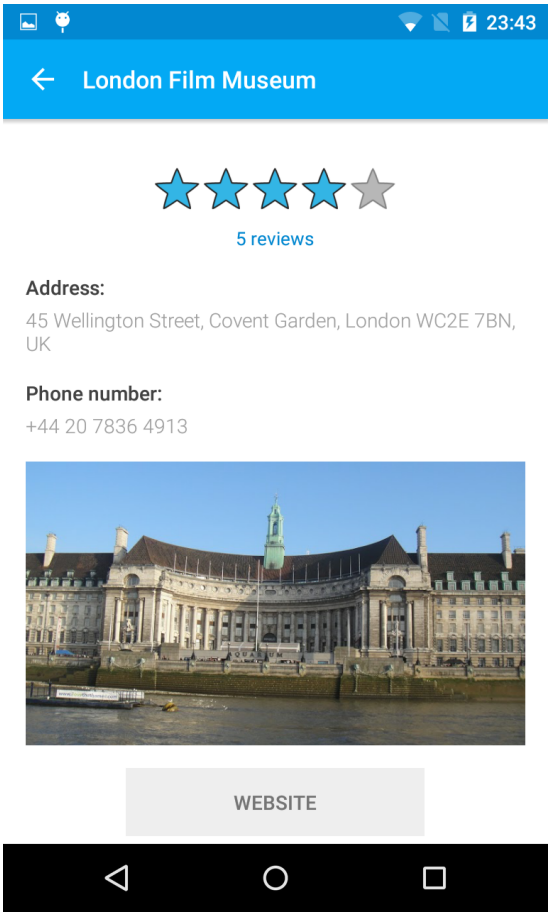
4.2 Ukážka užívateľského rozhrania

Ako hlavná téma mobilnej aplikácie bola zvolená modrá material farba[3] spolu s jej 'dark' verziou[4], ktorá podľa môjho názoru pôsobí pozitívne na cieľového užívateľa. Na texty sú použité prevažne dva typy sivej material farby[5]. Ukážku užívateľského rozhrania je možné vidieť na obrázkoch 13 a 14.

[3]Light Blue(500): #03A9F4 viz. https://material.google.com/style/color.html#color-color-palette
[4]Light Blue(700): #0288D1
[5]Grey(500): #9E9E9E a Grey(700): #616161



Obrázek 13: UI: Obrazovka 'Home'



Obrázek 14: UI: Obrazovka detailu POI

5 Implementácia

Implementácia prebiehala postupne podľa analýzy riešenia. Na začiatku bolo nutné naštudovať API z ktorého bude aplikácia získavať dáta (v našom prípade webová služba Places). Bolo nutné zistiť:

- spôsoby komunikácie, ktoré webová služba podporuje a vybrať z nich jeden vhodný
- formát dát, s ktorým služba pracuje
- typy parametrov, ktoré podporuje
- formát odpovede, ktorú služba vracia a s ktorou bude aplikácie finálne pracovať
- typy a rozsah dát, ktoré služba vracia

Po splnení tejto časti sa bolo nutné zamyslieť nad spôsobom akým spracovať jednotlivé získané body záujmu s cieľom určiť ich 'zaujímavosť' a ponúknuť užívateľovi len tie 'najzaujímavejšie'. K tomu som sa rozhodol použiť online voľne dostupný zdroj informácii vo forme wikipédie z ktorej som zisťoval množstvo informácií vzťahujúcich sa k práve analyzovanému bodu záujmu.
Po dokončení tejto úlohy bol algoritmus získavania bodov záujmu pre užívateľa hotový a následne bolo nutné sa zamerať na spôsob 'cachovania' týchto POI (Point of Interest) k tomu aby boli dostupné offline. Rozhodol som sa opäť využiť priamo samotný systém Android a jeho vstavanú databázu SQLite, ktorá je vhodným kandidátom na lokálny 'storing' štruktúrovaných dát (v našom prípade bodov záujmu).

Následne bolo nutné zistiť a implementovať algoritmus pre dynamické generovanie otázok vzťahujúcich sa k daným bodom záujmu. K tomuto účelu bolo použité API 'wikitrivia', ktoré je napísané v programovacom jazyku *Python*. Algoritmus využíva na generovanie otázok NLP (Natural Language Processing) spolu s podpornými Python knižnicami.

Nakoniec sa do aplikácie implementovali posledné funkcie ako možnosť prezrieť si reviews online jednotlivých bodov záujmu (v prípade, že sú dostupné), uživatelské questy a zdieľanie informácií v rámci aplikácie na sociálnej sieti.

5.1 Zdrojový kód

Od samotného začiatku návrhu a implementácie projektu, ktorý je predmetom tejto práce bol kladený dôraz na to, aby bola architektúra projektu napísaná rozumne, čisto, s istou dávkou abstrakcie a pomocou vopred doporučených postupov v rámci konkrétného kontextu (v našom prípade platformy Android). Ako výsledok dáva navrhnutá architektúra následujúce benefity:

- Zdrojový kód je čistý a čitateľný
- Jednoduchšie a rýchlejšie aktualizovateľný
- Aplikačná logika je oddelená od vzhľadu
- V kóde sa nenachádzajú redundantné funkcie a metódy
- Menej kódu = menej chýb
- Dodžiavané 'name conventions' a okomentovaný kód sú významné v prípade ďalšieho vývojára ale aj pre samotného autora v prípade, že sa k projektu vráti po dlhej časovej prestávke.

5.1.1 Rozdelenie kódu

Vývoj pre platformu Android (rovnako ako v čistej Jave) umožňuje rozdeliť zdrojový kód do logicky oddelených častí nazývaných **packages**. Aj keď to nie je žiadne písané pravidlo je to veľmi výhodné a prispieva to k omnoho lepšej a rýchlejšej orientácií v projekte. Obzvlášť je to 'good practice' v prípade veľkého projektu kde sa vývojár bez toho nezaobíde obzvlášť ak pracuje v tíme.

Mobilná aplikácia má následujúce rozdelenie kódu:

- com.gcoach.android.app.adapter
- com.gcoach.android.app.async
- com.gcoach.android.app.client
  - request
    - google
    - wiki
  - response
    - google
    - wiki
- com.gcoach.android.app.common
- com.gcoach.android.app.database
- com.gcoach.android.app.exception
- com.gcoach.android.app.helper
- com.gcoach.android.app.model
- com.gcoach.android.app.service
- com.gcoach.android.app.ui
  - activity
  - fragment
    - dialog
- com.gcoach.android.app.utility
- com.gcoach.android.app.view
- com.gcoach.android.app.AppController

Názvy jednotlivých packages (balíčkov) sú zvolené tak aby jednoznačne popisovali účel konkrétneho balíčku aby keď sa nato pozrie nezainteresovaný človek vedel jednoznačne povedať k čomu slúži / bude slúžiť daný balíček.

V následujúcom texte popíšem jednotlivé balíčky použité v mobilnej aplikácii a podám dôvody prečo som ich vytvoril a použil. Od rozumného zvolenia a rozdelenia jednotlivých balíčkov dosť závisí celková čitateľnosť kódu a prehľadnosť projektu ako bolo spomenuté na začiatku sekcie. Je dobré to pripomenúť ešte raz pretože mnoho vývojárov nato zabúda.

5.1.2 Balíček adapter

Balíček obsahuje implementáciu adaptérov použitých v zoznamoch dát (Collection). Rozhodol som sa pre vlastnú implementáciu adaptérov pretože Android síce ponúka už viacmenej hotové adaptéry ako *ArrayAdapter* napríklad ale tie sú značne limitované a hlavne neimplementujú návrhový vzor *ViewHolder*, ktorý je veľmi dôležitý z hľadiska práce s pamäťou a celkovo výkonnosťou adaptéru. Funkčnosť daného návrhového zdroju spočíva v tom, že v prípade, že užívateľ 'skroluje' napríklad smerom dole a následne naspäť hore tak v tomto prípade funguje *ViewHolder* ako cachovací mechanizmus a riadky, ktoré mobil postupným skrolovaním znova zobrazí užívateľovi sa znova nevytvárajú ale len sa *recyklujú*.

5.1.3 Balíček async

Balíček obsahuje implementáciu asynchronných workerov použitých v aplikácii. Asynchronný worker je zjednodušene trieda, ktorá v sebe zapúzdruje funkcionalitu spustenia kusu kódu na vlákne na pozadí, ktorý sa spúšta asynchrónne teda nezávisle. Tento spôsob sa využíva aby aplikácia fungovala plynuje ale zároveň dokázala na pozadí spracovávať nejaké úlohy a podľa potreby informovať užívateľa prostredníctvom definovaného *callbacku*, ktorý sa volá na UI vlákne (z ktoré jediného je možné manipulovať s UI prvkami). V súčasnej verzii aplikácie tento balíček obsahuje workera, ktorý sa stará o generovanie obrázku, ktorý užívateľ môže zdieľať so svojmi priateľmi.

5.1.4 Balíček client

Balíček zapúzdruje celkovú logiku komunikácie aplikácie so zdrojmi dát. Obsahuje 'subpackages' **request** a **response** ktoré implementujú spracovávanie jednotlivých *HTTP* dotazov a odpovedí. Práve na tomto mieste sú naimplementované všetky dotazy a odpovede s ktorými aplikácia pracuje a získavá dáta z webových služieb. Všetky definované requesty dedia z 'base requestu' rovnako ako všetky odpovede dedia z 'base response'. Tento spôsob návrhu je veľmi užitočný pretože sa týmto vyhneme písaniu redundantného kódu v momente keď budeme chcieť napríklad implementovať podporu indikácie chybovej odpovede z webovej služby. Týmto spôsobom to stačí implementovať len v baseovom 'requeste' a vec je hotová. Podobne to platí aj pre odpovede. Komplexnejší *REST* server vracia niekoľko atribútov spoločných pre každý request napríklad timestamp, response code a pod. Tieto atributy stačí namapovať do baseovej triedy pre odpoveď a nemusia sa písať do každej triedy reprezentujúcej odpoveď z nejakého endpointu.

5.1.5 Balíček common

Balíček obsahuje 'spoločné' veci, ktoré sa používajú v rámci celého projektu. Na tomto mieste sú definované konštanty, enpointy, credentials a ďalšie veci, ktoré sa používajú v rámci celej aplikácie. Z toho balíčka by som bližšie opísal triedu *Fonts*, ktorá podľa potreby načítava a následne cachuje font používaný v mobilnej aplikácii.
Na cachovanie fontu a jeho rezov je použitá trieda *LruCache*, ktoré je na cachovanie primárne navrhnutá a optimalizována pre platformu Android.

5.1.6 Balíček database

Balíček obsahuje implementáciu databázy a prístup k nej prostredníctvom takzvaných *mapperov*. Všetky objekty, ktoré akýmkoľvek spôsobom spolupracujú s databázou sú výhradne v tomto balíčku ako logicky prepojení spolupracovníci. Kedže mimo tento balíček nie je dôvod, aby mal akýkoľvek objekt prístup k parametrom databázy ako napríklad k jej stĺpcom alebo k názvom tabuliek tak sú všetky parametre definované ako chránené tzv. *protected* aby boli skutočne dostupné a viditeľné len v rámci tohto balíčku. Je dobré poznamenať, že *Java* umožnuje získať prístup k private a protected parametrom prostredníctom *Reflection* ale to je vec, ktorá by sa používať nemala pretože porušuje prvé pravidlo objektovo orientovaného prístupu k programovaniu a to je zapúzdrenie. Aj keď sa nájdu prípady kedy nie je iná možnosť a je nutné použiť spomenutú *Reflection* metódu ale v rámci tejto aplikácie ten prípad nenastal. S databázou komunikujú mappery, ktoré v sebe zapúzdrujú základné *CRUD* operácie (Create, Read, Update, Delete).

5.1.7 Balíček exception

Balíček obsahuje vlastnú implementáciu výnimiek použitých v projekte. Implementácia vlastných výnimiek je veľmi výhodná najmä vo väčšom projekte kedy môžeme veľmi detailne určiť typ výnimky a tak presne zistiť v akej časti projektu nastáva problém. V tejto aplikácii je v aktuálnej verzii použitá jedna výnimka určujúca problém pri analýze dát napríklad v dôsledku problému konektivity.

5.1.8 Balíček helper

Balíček obsahuje pomocné *callbacky* používané v rámci projektu na komunikáciu medzi objektami všeobecne. Konkrétne v mobilnej aplikácii balíček obsahuje vlastné *callbacky*, ktoré sú použité na komunikáciu koreňovej obrazovky tzv. *FragmentActivity* so sekciami, ktoré su reprezentované *fragmentami* a komunikácie medzi adaptérmi a ich hostiteľmi (*fragment activities*). *Callbacky* sú definované ako rozhrania pretože každý fragment by mal mať referenciu výhradne len na interface, ktorý implementuje jeho 'container'. Týmto sa zaručí znovupoužiteľnosť daného rozhrania aj s iným fragmentom, ktorý obsahuje podobnú funkcionalitu.

5.1.9 Balíček model

Balíček obsahuje modely objektov použitých v projekte. Konkrétne v rámci tejto aplikácie sa jedná o reprezentáciu bodu záujmu, questu spojeného s bodom záujmu, profilom ktorý reprezentuje užívateľa a dynamicky generovanej otázky.

5.1.10 Balíček service

Balíček obsahuje imlementáciu služieb použitých v projekte a na komunikáciu s internetom (Networking). Pre každú použitú službu je vytvorená *service* trieda, ktorá sa predáva do manažéra networkingu, ktorý je v tomto prípade *Robospice* s modulom *Retrofit*. Jednotlivé služby slúžia na definovanie base *URL* serveru, konvertera odpovede a celkovo na customizáciu práce s webovou službou. Jednotlivé *HTTP* služby sú používané balíčkom **client.request**.

5.1.11 Balíček client.request a client.response

Balíček obsahuje implementáciu jednotlivých HTTP dotazov a odpovedí. Tento balíček má ešte sub balíčky pomenované ako **google** a **wiki**. Ako už názvy hovoria, v sub balíčku **google** sa nachádzajú dotazy a odpovede spojené s webovou službou Google Places a v sub balíčku **wiki** s webovou službou Wikipédie.

5.1.12 Balíček ui

Balíček zapúzdruje kompletnú implementáciu užívateľského rozhrania. Obsahuje konkrétne sub balíček **activity**, kde sa nachádzajú všetky 'activity' používané mobilnou aplikáciou a sub balíček **fragment**, kde sa nachádzajú všetky 'fragmenty' používané aplikáciou. Sub balíček **fragment** ešte obsahuje ďalší sub balíček **dialog**, kde sa nachádzajú všetky 'fragment dialógy' použité v rámci aplikácie. Ku všetkým trom komponentám sú vytvorené abstraktné basové triedy.

5.1.13 Balíček utility

Balíček obsahuje rôzne 'utility' metódy a funkcie použité v rámci projektu. Cieľom tohto balíčku je odstrániť redundantný kód z projektu. Inými slovami sa v tomto balíčku nachádzajú triedy, ktoré obsahujú statické metódy používané v rámci celej aplikácie. Sú to metódy, ktoré sa používajú na viacerých miestach v aplikácii ale sú definované len na jednom mieste a tým pádom nie su redundantné a jednoduchšie aktualizovateľné. Zmena na jednom mieste sa reflektuje všade kde je daná metóda použitá.

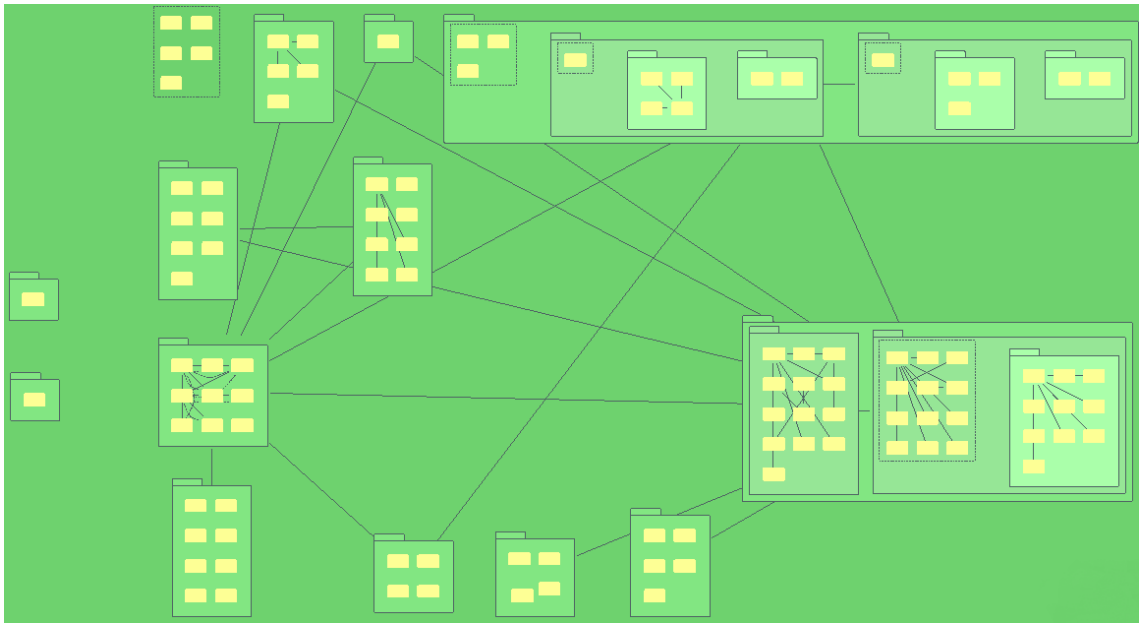
5.1.14 Balíček view

Balíček obsahuje implementáciu vlastných (custom) widgetov, ktoré boli potrebné pre aplikáciu. Síce Android ponúka veľké množstvo rôznych widgetov ale nie vždy to stačí. V našom prípade bolo nutné definovať vlastný textový widget s vlastným fontom (v aplikácii je použitý font *Roboto*), vlastný widget reprezentujúci kompas a vlastný widget pomenovaný ako *TabBar*, ktorý je použitý ako hlavná (spodná) navigácia mobilnej aplikácie.

5.1.15 AppController

Balíček predstavuje **Application** komponentu, ktorá je inicializovaná ako prvá v rámci životného cyklu aplikácie a je určená na inicializáciu objektov, ktoré musia byť vzhľadom na 'lifecycle' aplikácie inicializované na tomto mieste. Na tomto mieste sa zvyčajne inicializujú externé knižnice ako napríklad Facebook SDK, ktoré je v mobilnej aplikácii použité.

Na obrázku 15 je ukážka UML diagramu predstavujúca *package view* hierarchiu mobilnej aplikácie.



Obrázek 15: Ukážka Package View UML diagramu

V diagrame nie sú z hľadiska zachovania prehľadnosti uvedené texty. Kompletný diagram je vyexportovaný v *JSON* formáte a je dostupný v prílohe práce na CD.

5.2 Získavanie a analýza bodov záujmu

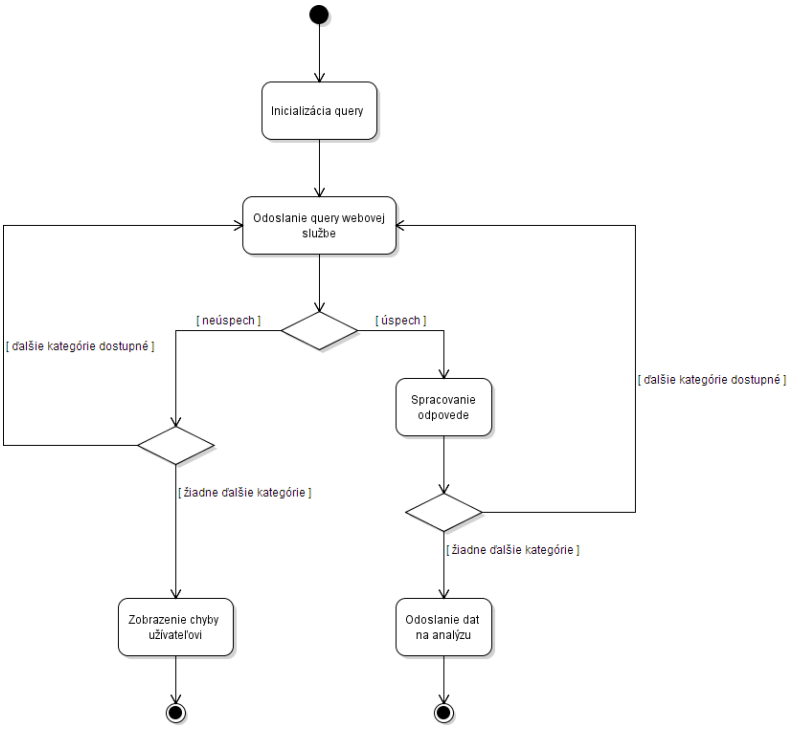
Dôležitou časťou aplikácie ja algoritmus získavania a analýzy bodov záujmu, ktoré aplikácia ponúka užívateľovi. Ako zdroj dát je použitá webová služba *Google Places*, ktorá prostredníctvom komunikácie cez špecifické *http queries* vracia jednotlivé dáta bodov záujmu. Vo výpise 4 sa nachádza príklad takejto query.

```
https://maps.googleapis.com/maps/api/place/nearbysearch/json?language=en&
    radius=1000&location=51.5088345,-0.1219935&key={api_key}&type=museum
```

Výpis 4: Google Places query URL

Ako je zrejmé z výpisu, touto query aplikácia 'požiada' webovú službu aby jej vrátila anglicky (ak je preklad dostupný) body záujmu v rádiuse 1000 metrov v okolí užívateľa, ktoré je definované aktuálnou polohou zariadenia prostredníctvom zemepisnej šírky a dĺžky a aby dané body boli múzeá.

Tok komunikácie s webovou službou *Google Places* sa náchádza na obrázku 16 reprezentovanom diagramom aktivít.



Obrázek 16: Diagram aktivít komunikácie s webovou službou Google Places

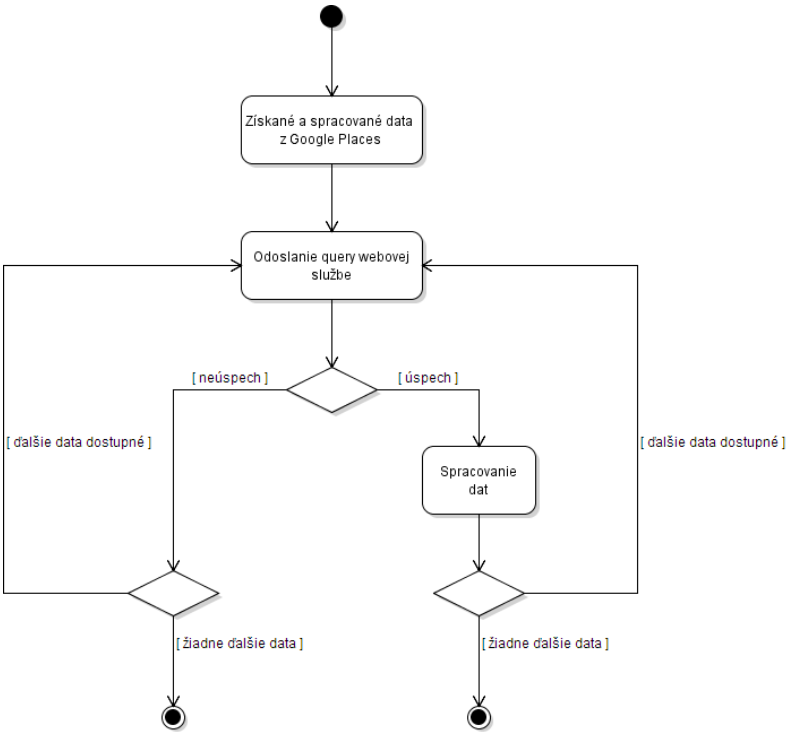
V momente kedy sú body záujmu obdržané z webovej služby prichádza na radu ich analýza a určenie ich 'zaujímavosti'. K tomuto je v mobilnej aplikácii použitá webová služba *wikipédie*, ktorá podobne ako webová služba *Google Places* umožnuje získavať dáta na základe špecifickej *http query*. Príklad konkrétnej *query* je vo výpise 5.

```
https://en.wikipedia.org/w/api.php?action=parse&format=json&prop=text&page={
    search_phase}
```

Výpis 5: Wikipédia query URL

Pre účely mobilnej aplikácie sa používa endpoint webovej služby *wikipédie*, ktorý vráti sparsovanú 'stránku' asociovanú s vyhľadávanou frázou, ktorá je v tomto prípade názov bodu záujmu. Následne sa odpoveď vrátená webovou službou spracuje, získa sa z nej veľkosť a na základe veľkosti odpovede aplikácia určí 'zaujímavosť' konkrétneho bodu záujmu. Algoritmus je v tomto prípade priamočiary. Čim väčšiu odpoveď vráti služba, tým existuje viac referencií a dát asociovaných ku konkrétnemu bodu záujmu a tým je 'dôležitosť' bodu záujmu väčšia.

Tok komunikácie a webovou službou *Wikipédie* je vyobrazený na obrázku 17.



Obrázek 17: Diagram aktivít komunikácie so službou Wikipédia

5.3 Perzistencia bodov záujmu

Ako formát na perzistenciu dát bodov záujmu bola na základe analýzy zvolená *SQLite* databáza systému Android.

5.3.1 SQLite databáza

V krátkosti je to kompaktný databázový engine, ktoré nemá narozdiel od väčšiny ostatných databáz separátny server ale číta a zapisuje priamo na disk [11]. Je ideálna pre mobilné zariadenia pretože so všetkými zapnutými funkciami jej veľkosť nepresiahne 500 KB [11]. Pre potreby vyvíjanej mobilnej aplikácie je plne dostačujúca.

Na aplikačnej vrstve je databáza implementovaná ako *Singleton*.

5.3.2 Schéma

Databáza použitá v mobilnej aplikácii obsahuje celkovo 3 tabuľky, ktoré reprezentujú:

- Bod záujmu pomenovanú ako **places**
- Quest spojený z bodom záujmu pomenovanú ako **quests**
- Užívateľa pomenovanú ako **users**

Na obrázkoch 18, 19 a 20 sa nachádzajú databázové schémy jednotlivých tabuliek.

Tabuľka *places* obsahuje 10 stĺpcov, ktoré predstavujú jednotlivé vlastnosti bodu záujmu a ktoré aplikácia používa. Patria sem napríklad súradnice, názov a identifikátory, ktoré jednoznačne určujú každý bod záujmu. Ako primárny kľúč je použitý *hash code* identifikátora bodu záujmu získaný z jeho textovej reprezentácie.

Tabuľka *quests* obsahuje 5 stĺpcov, ktoré špecifikujú výsledok questu splneného užívateľom. Aktuálne sú to čas, dĺžka, vzdialenosť questu, ktorú musel užívateľ precestovať aby daný quest splnil, identifikátor bodu záujmu, ktorý predstavuje cudzí kľuč k tabuľke *places* a identifikátor questu, ktorý je automaticky generovaný databázou a predstavuje primárny kľúč.

Tabuľka *profiles* reprezentuje používateľa aplikácie. Obsahuje základné informácie o užívateľovi akými sú napríklad meno a e-mailová adresa.

Vo výpise 6 sa nachádza ukážka SQL dotazu pre vytvorenie tabuľky *places*.

	id	non_hashed_id	place_id	name	icon	vicinity	lat	lng	types	openNow
	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	2035878805	56d38a949151545l	ChIJZ4MM0TQbdkI	Freemasons' Hall	https:// maps.gstatic.com/I	60 Great Queen Street, London	51.5149571	-0.1211454	library,point_of_intl	0
2	1937707367	4092c38d954e1a9l	ChIJV1jTQrUEdkgI	British Library of Political and Econl	https:// maps.gstatic.com/I	10 Portugal Street, London	51.51469	-0.115698	library,point_of_intl	1
3	1917053655	6ebed302066d105l	ChIJH-tBOc4EdkgRlsJgzI	St Martin-in-the-Fields	https:// maps.gstatic.com/I	Trafalgar Square, London	51.5087908	-0.1267527	church,place_of_wl	0
4	1616595641	ded3d4530b95071l	ChIJq4lX1doEdkgI	Churchill War Rooms	https:// maps.gstatic.com/I	Clive Steps, King Charles Street, Lonl	51.5021538	-0.1292207	museum,point_of_il	1
5	1602170522	98bbc627e6dfab2fl	ChIJr8J75s0Edkgl	London	https:// maps.gstatic.com/I	London	51.5102003	-0.1280532	library,point_of_intl	NULL
6	1546155556	a4bd7602f14eef2fel	ChIJaQQ3lEobdkgI	Sir John Soane's Museum	https:// maps.gstatic.com/I	13 Lincoln's Inn Fields, London	51.5170382	-0.1174699	museum,point_of_il	0
7	1329165786	9b380e134dd640el	ChIJ03seTNlEdkgI	Westminster Reference Library	https:// maps.gstatic.com/I	35 Saint Martin's Street, London, Lol	51.5096489	-0.1298028	library,point_of_intl	0
8	1173788043	ab68ee8eb42d51el	ChIJZf2Jlc4EdkgRl	The Courtauld Gallery	https:// maps.gstatic.com/I	Somerset House, Strand, London	51.5115744	-0.1176965	art_gallery,store,cl	1
9	971853061	1390c6eda79aaacl	ChIJG7j3T8oEdkgI	London Transport Museum	https:// maps.gstatic.com/I	Covent Garden Piazza, London	51.5119271	-0.1214268	museum,point_of_il	1

Obrázek 18: Ukážka schémy tabuľky bodov záujmu

Na vyššie uvedenom obrázku sa nachádza náhľad databázovej schémy tabuľky bodov záujmu. Ako bolo vyššie spomenuté obsahuje celkovo 10 stĺpcov z ktorých sú pre aplikáciu najdôležitejšie stĺpce **lat**, **lng** reprezentujúce zemepisnú šírku a dĺžku polohy bodu záujmu, **name** reprezentujúci názov bodu záujmu a **id**, ktoré slúži ako jednoznačný identifikátor bodu záujmu a je použitý ako primárny kľúč tabuľky.

	id	distance	duration	date_time	place_id
	Filter	Filter	Filter	Filter	Filter
1	1	0.32624493426963	14	2016-06-26 18:57:28	-1556402990
2	2	0.16656122880394	4	2016-06-26 18:57:42	-1831421100

Obrázek 19: Ukážka schémy tabuľky questov

Na obrázku je vidieť schéma tabuľky questov (úloh). Celkovo tabuľka obsahuje 5 stĺpcov, ktoré sú všetky dôležité. Stĺpec **id** je auto incrementovaný primárny kľúč, **placeId** predstavuje cudzí kľúč, ktorý je referenciou na primárny kľúč tabuľky *places*. Zvyšné stĺpce predstavujú dĺžku trvania questu a vzdialenosť, ktorú užívateľ absolvoval s cieľom spĺniť daný quest.

	id	firstname	lastname	email	age
	Filter	Filter	Filter	Filter	Filter
1	1	Simon	Dorociak	s.dorociak@gmail.com	24

Obrázek 20: Ukážka schémy tabuľky profilov

Tabuľka profilov obsahuje taktiež 5 stĺpcov kde sa zaznamenávajú základné informácie o uživateľovi používajúcom aplikáciu a to je jeho meno, priezvisko, e-mail a vek.

```
create table if not exists places (
   id integer primary key not null,
   non_hashed_id text not null,
   place_id text not null,
   name text not null,
   icon text not null,
   vicinity text not null,
   lat real not null,
   lng real not null,
   types text not null,
   openNow integer
);
```

Výpis 6: Ukážka SQL dotazu pre tabuľky places

Vo vyššie uvedenom výpise sa nachádza ukážka vytvorenia tabuľky **places** kde je možné vidieť, že id bude predstavovať primárny klúč tabuľky. V rámci aplikácie je to hashované id bodu záujmu (v tabuľke sa uchováva aj jeho originálna podoba vo forme stringu). Všetky ostatné atribúty okrem atribútu *openNow* sú definované ako *not null* čo znamená, že vždy musia mať priradenú nejakú hodnotu. Atribút *openNow* je nepovinný.

Rád by som ešte poznamenal, že samotný Android neobsahuje žiadny tool, ktorým by bolo možné si prezrieť databázové schémy. Z tohto dôvodu som pre účely tejto práce použil šikovný nástroj *DB Browser for SQLite* pomocou ktorého je možné si jednoducho a rýchlo prezrieť, overiť databázové schémy a urobiť si z nich napríklad screenshoty. Má svoje oficiálne stránky [6] ale je aj verejne dostupný na githube [7]. Ešte by som poznamenal, že je to klasická desktopová aplikácia.

5.3.3 Prístup a dotazy

Prístup k databáze je veľmi dôležitá vec z pohľadu plynulého behu aplikácie (užívateľského rozhrania). Z tohto dôvodu je nutné komunikovať s databázou výhradne v samostatnom vlákne aby nevznikol typický problém a to je 'freezing' aplikácie. V mobilnej aplikácii je použitý *asynchronný* prístup k databáze, ktorý neblokuje hlavné UI vlákno a zabezpečuje plynulý chod aplikácie. Konkrétne je v projekte použitá trieda *AsyncTask* ktorá zapúzdruje danú logiku a zároveň umožňuje komunikáciu s UI vláknom, ktorá je vo väčšine prípadoch využívaná na informovanie užívateľa 'o aktuálnom stave' vykonávanej práce.

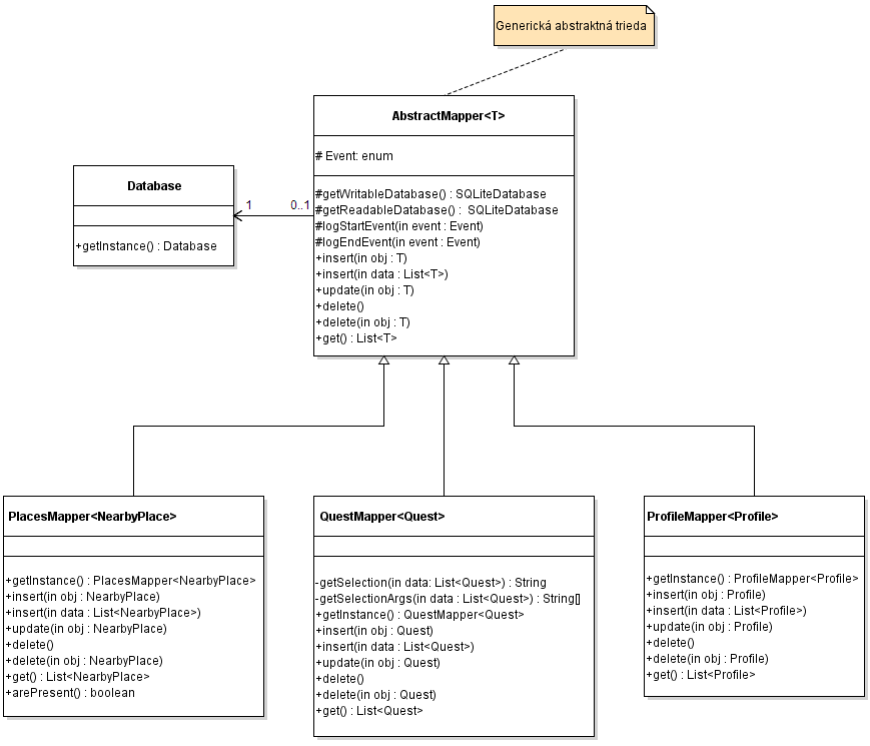
[6] http://sqlitebrowser.org/
[7] https://github.com/sqlitebrowser/sqlitebrowser

Samotnú komunikáciu s instanciou databázy majú na starosť jednotlivé databázové *mappery*, ktoré zapúzdrujú a implementujú základné CRUD operácie nad databázou a len prostredníctvom nich komunikuje aplikácia s databázou. Na obrázku 21 sa nachádza triedny diagram použitých mapperov.

Mobilná aplikácia používa pri komunikácii s databázou jednoduché SQL dotazy. Príklady sú uvedené vo výpise 7. Všetky SQL dotazy sú parametrizované. Konkrétne pre Android to znamená, že sa do SQL dotazu pridajú takzvané *placeholdery*, ktoré sú v runtime nahradené konkrétnymi hodnotami.

Táto technika má svoje výhody:

- Nevznikajú príliš dlhé dotazy a dotazy sú oveľa lepšie čitateľnejšie
- Dotazy môžeme považovať za bezpečné



Obrázek 21: Diagram tried databázových mapperov

Na uvedenom obrázku je ukážka diagramu tried databázových mapperov. V diagrame je vidieť, že všetky mappery použité v aplikácii dedia z *AbstractMapper<T>*, ktorý predstavuje abstraktnú generickú triedu z ktorej dedia všetky ďalšie mappery. Jedine tento mapper má väzbu na instanciu databázy pomocou chránených (*protected*) metód *getWriteableDatabase()* a

getReadableDatabase() ku ktorým ma prístup každý jeho potomok (subclass). Triedu som spravil za využitia generík aby bola daná abstraktná trieda použiteľná pre akýkoľvek objekt.

```
select * from quests
select * from places where places.id in (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?,
    ?, ?, ?, ?, ?, ?, ?)
```

Výpis 7: Príklad SQL datazov nad databázou

Z výpisu *SQL* dotazov by som rád bližšie vysvetlil uvedené otázniky. Ako bolo spomenuté vyššie tak aplikácia používa parametrizované dotazy. Na Androide fungujú tak, že sa namiesto konkrétnych hodnôt vložia do dotazu otázniky kde počet otáznikov = počet hodnôt/parametrov. Tieto otázniky sú následne kódom bezpečne nahradené dosadenými hodnotami ako je vidieť vo výpise 8.

```
List<NearbyPlace> places = new ArrayList<>();
 _c = db.query(Database.TABLE_PLACES, null, getSelection(result),
    getSelectionArgs(result), null, null, null);
if (_c != null && _c.moveToFirst()) {
   do {
      NearbyPlace item = new NearbyPlace();
      item.setHashedId(_c.getInt(_c.getColumnIndex(Columns.ID)));
      ...
      places.add(item);
   } while (_c.moveToNext());
}
```

Výpis 8: Príklad spustenia parametrizovaného dotazu kódom

V ukážke kódu je vidieť naplnenie databázového kurzora datami z databázy. Metóda *getSelection()* vracia vyššie uvedené dynamicky vygenerované otázniky ako parameter metódy *query*, ktoré budú pred jej spustením nahradené datami, ktoré vracia metóda *getSelectionArgs()*. Vždy platí, že počet otáznikov (placeholderov) = počet parametrov (args).

5.3.4 Offline výpočet vzdialenosti bodu záujmu

Jedná sa o offline výpočet vzdialenosti konkrétného bodu záujmu od polohy mobilného zariadenia (v online režime sa o to stará samotná webová služba).

Tento výpočet sa používa v prípade, že užívateľ spustil aplikáciu v offline režime a aplikácia musí uživateľovi zobraziť body záujmu získané z lokálnej databázy.
Aby sa zachovala rovnaká funkcionalita a užívateľovi sa zobrazili skutočne len tie body záujmu, ktoré sú v jeho blízkom okolí aplikácia používa na určenie vzdialenosti dvoch bodov na Zemi *haversinovu formulu* pre výpočet vzdialenosti ktorá má následujúci tvar [12, 13]:

$$a = \sin^2(\Delta\varphi/2) + \cos\varphi_1 * \cos\varphi_2 * sin^2(\Delta\lambda/2) \tag{1}$$

$$c = 2 * \text{atan2}(\sqrt{a}, \sqrt{1-a}) \tag{2}$$

$$d = R * c \tag{3}$$

Kde φ je zemepisná šírka, λ je zemepisná dĺžka, R je zemský stredný polomer (6371 km), $\Delta\varphi$ a $\Delta\lambda$ sú vzdialenosti medzi bodmi. Zemepisné šírky a dĺžky sú v radiánoch.

5.4 Dynamické generovanie otázok

Ďalšia dôležitá funkcia mobilnej aplikácie je schopnosť dynamicky generovať otázky asociované s konkrétnym bodom záujmu. K tomuto účelu aplikácia používa knižnicu *wikitrivia*, ktorá implementuje algoritmus využívajúci NLP (Nature Language Processing) na dynamické generovanie otázok. Knižnica je napísaná v programovacom jazyku *Python*.

Kedže systém Android neponúka žiadny *rozumný* spôsob komunikácie a spustenia skriptu napísanom v jazyku *Python* tak bola pre potreby mobilnej aplikácie vytvorená *REST webová služba*, ktorá implementuje spustenie skriptu a následne spracované dáta (v tomto konkrétnom príde sa jedná o otázky) vracia aplikácii prostredníctvom HTTP dotazu.

5.4.1 Knižnica wikitrivia

Knižnica predstavuje API napísané v jazyku *Python*, ktoré mobilná aplikácia používa na dynamické generovanie otázok vzťahujúcich sa ku konkrétnym bodom záujmu. K danej funkcionalite využíva NLP (Natural Language Processing) za použitia TextBlobu, NLTK (Natural Language Toolkit) a WordNetu.

TextBlob je knižnica pre jazyk *Python* určená pre spracovávanie textových dát. Poskytuje API na bežné úlohy NLP ako je 'part-of-speech tagging', extrakcia podstatných mien, klasifikácia alebo preklad [15]. V knižnici použitej mobilnou aplikáciou je použitá na extrakciu viet z článkov wikipédie.

NLTK je komplexná a najpopulárnejšia knižnica pre NLP pre jazyk *Python* z ktorej je TextBlob len jednoduchý 'wrapper'.

Generovanie otázok prebieha podľa algoritmu viz. výpis 9.

```
def evaluateSentence(sentence):
if (startWithAdverb or sentenceLength(sentence.words) < 6) {
   return null;
}
replace_nouns = []
similar_words = []
for word, tag in sentence.tags {
   if (!nonProperNounExistInArticleTitle(tag, word)) {
      for phase in sentence.noun_phrases:
         if (startsWithApostrophe(phase[0]) break;
         if word in phrase:
            removeLastTwoWordsInPhrase(replace_nouns, phrase) break;
      if (replace_nouns.length == 0) {
         replace_nouns.append(word)
      }
   }

if (replace_nouns.length == 0) return null;
if (replace_nouns.length == 1) {
   similar_words = getSimilarWords(replace_nouns[0]);
}
return gerNormalizedSentence(sentence);
}
```

Výpis 9: Algoritmus generovania otázok

Podľa vyššie uvedeného pseudo kódu algoritmus funguje následovne:

- Ak daná veta začína príslovkou alebo má menej ako 6 znakov tak algoritmus predpokladá, že nebude dobrým kadidátom na výslednú otázku.
- Následuje cyklus v ktorom sa odstránia podstatné mená, ktoré sa nenachádzajú v názve článku.
- Pokiaľ je aktuálna fráza vhodná, odstránia sa posledné dve slová z frázy.
- Pokiaľ fráza začína apostrofom tak to algoritmus ignoruje.
- Nakoniec sa odstránia spomenuté posledné dve slová z frázy

- Ak sa nenašlo v žiadnej fráze slovo tak sa použije vstup
- Ak sa nenašli žiadne slová na nahradenie tak metóda nevracia nič.
- Ak sa našlo jedno slovo tak sa použije WordNet na získanie podobných slov. Iný výsledok sa v algoritme ignoruje.
- Nakoniec sa z pôvodnej vety odstránia odstránené slová a nahradia sa v aktuálnej implementácii medzerou tvorenou podtržítkom a metóda vracia upravenú vetu, ktorá bude použitá ako otázka.

Ako som spomenul na začiatku tak vety sa získavajú z článkov wikipédie pomocou TextBlobu viz. výpis 10.

```
summary = TextBlob(wikipedia.page(title).summary)
for sentence : summary.sentences:
   // analysing each sentence
```

Výpis 10: Algoritmus získavania viet z wiki článkov

Inicializuje sa *summary* objekt pomocou TextBlobu pre aktuálny nadpis (title), ktorý obsahuje jednotlivé vety, ktoré sú algoritmom uvedeným vo výpise 9 spracovávané.

Podobné slová sa získavajú pomocou WordNetu čo je v krátkosti lexikálna databáza pre anglický jazyk [17]. Pseudo kód použitého algoritmu je vo výpise 11

```
def getSimilarWords(self, word)
   synsets = getWordnetSynsets(word)
   if (synsets.length == 0) {
      return [];
   } else {
       synset = synsets[0]
   }
   hypernym = getHypernymFromSynset()
   hyponyms = hypernym.getHyponyms()
   similar_words = getSimilarWordsFromHyponyms(hyponyms)
   return words;
```

Výpis 11: Algoritmus získavania podobných slov

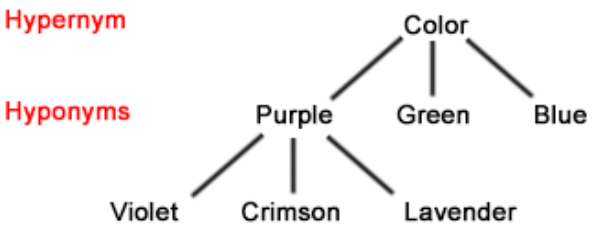
Podľa pseudo kódu 11 algoritmus funguje následovne:

- Pomocou WordNetu sa získajú synsety pre hľadané slovo
- Pokiaľ sa nenašiel žiadny synset tak sa vracajú podobné slová ako prázdne pole
- Ak sa našiel aspoň jeden synset tak súčasný algoritmus použije vždy prvý synset
- Následne sa získa zo synsetu hypernym (vždy sa berie prvý podobne ako to je pri synsete)
- Z hypernymu sa získajú hyponymy
- Z hyponymov sa použije prvých 8 nájdených a tie metóda vráti ako pole podobných slov

Nakoniec by som ešte stručne vysvetlil pár výrazov:

- **synset** je kolekcia synoným do ktorej sú grupované podstatné mená, slovesá, prídavné mená a príslovky [16]
- **hypernym** je slovo alebo fráza, ktorého sémantická časť je obsiahnutá v nejakom inom slove a následne **hyponym** je slovo alebo fráza, ktorého sémantická časť je viac špecifická, konkrétnejšia ako u **hypernymu** [5, 6]

Myslím si, že pojmy ako je **hypernym** a **hyponym** lepšie vysvetlí obrázok 22.



Obrázek 22: Ukážka hypernymov a hyponymov

5.4.2 REST webová služba

Ako bolo spomenuté v úvode bolo nutné vytvoriť *REST službu*, ktorá bude implementovať spustenie skriptu v jazyku *Python* a výsledok skriptu predať mobilnej aplikácii. *REST* služba je naimplementovaná v jazyku *Java* v spolupráci so *Spring* frameworkom.
Samotná práca so *Spring* frameworkom je efektívna a veľmi kompaktná a pre účely mobilnej aplikácie bola ideálna. Služba ponúka dva endpointy:

- endpoint **wiki**
- endpoint **wikimultiple**

Endpoint **wiki** spúšťa skript pre vyhľadávaciu frázu, ktorá sa skladá z práve jedného slova. Endpoint **wikimultiple** funguje rovnako s tým rozdielom, že slúži na spustenie skriptu pre frázu, ktorá má viac ako jedno slovo.

Vo výpise 12 je ukážka implementácie endpointu. Algoritmus, ktorý implementuje endpoint funguje následovne:

- Spustí sa prostredníctvom príkazového riadku *Python* skript ako samostatný proces
- Odpoveď procesu sa postupne spracuje až pokiaľ sa proces automaticky neukončí (neskončí vykonávanie skriptu)
- Konečná odpoveď sa normalizuje pre potreby mobilnej aplikácie a 'odošle' do mobilného zariadenia.

```
@RequestMapping(value = "/wiki/{title}", method = RequestMethod.GET, headers =
    "Accept=application/json")
  public String getInfo(@PathVariable String title) {
     try {
        final StringBuilder builder = new StringBuilder();
        final Process proc = Runtime.getRuntime().exec("wikitrivia " + title);
        new Thread(new Runnable() {
           @Override
           public void run() {
              BufferedReader reader = new BufferedReader(new InputStreamReader(
                  proc.getInputStream()));
              String line = null;
              try {
```

```
                while ((line = reader.readLine()) != null) {
                    builder.append(line);
                }
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
    }).start();
    proc.waitFor();
    return getNormalizedResult(builder.toString());
} catch (Exception e) {
    return "";
}
}
```

Výpis 12: Príklad implementácie REST endpointu

Vyššie uvedená *REST* metóda funguje detailnejšie následovne. Prostredníctvom spring anotácie definujem názov endpointu **wiki**, ktorý má jeden parameter **title**. Nakoniec sa ešte definuje formát odpovede a to je *JSON*.
Následne sa cez príkazový riadok spustí ako samotný proces skript a počká sa na odpoveď. V samostatnom vlákne sa postupe spracujú dáta procesu a uložia sa do *StringBilderu* ako text. *StringBuilder* je pomocná trieda, ktorá slúži na manipuláciu s textom a mal by sa využívať v prípade generovania textu v cykle s cieľom zachovania výkonu.
Po prečítaní všetkých dát a ukončení procesu sa nakoniec získané textové dáta normalizujú (odstránia sa nadbytočné znaky a symboly, ktoré generuje skript v rámci svojho logovania) pre mobilnú aplikáciu a vrátia sa ako odpoveď do mobilnej aplikácie.

5.5 Problémy pri implementácii

Počas implementácie som sa stretol viacmenej len s jedným problémom a to bola integrácia dynamického generovania otázok. Kedže algoritmus je napísaný v jazyku *Python* nebolo ho možné priamo spustiť v mobilnej aplikácii. Problém bol vyriešený implementáciou vlastného *REST* servera, ktorý sa stará a spustenie skriptu a vrátenie jeho dát mobilnej aplikácii.

6 Testy a výsledky

Behom celého vývoja bola mobilná aplikácia priebežne testovaná, ladená aby sa predošlo hromadným opravám na konci vývoja. Testovanie prebiehalo väčšinou na reálnych zariadeniach. Emulátory boli z testovania takmer vypustené pretože sú to síce kópie jednotlivých mobilných zariadení ale nemožno na nich spraviť moc vierohodné testy kedže sú spustené na inom zariadení a ich výkon z väčšej časti závisí od daného zariadenia (notebook, desktop). S emulátormi sa testovalo len užívateľské rozhranie.
V rámci mobilnej aplikácie sa testovala kompletná analýza bodov záujmu s ich ukladaním do databázy a ich načítavanie z databázy v offline móde.

Ešte by som rád poznamenal, že v rámci práce sa mala aplikácia otestovať s vyšším počtom užívateľov a spracovať feedback. K tomuto nedošlo pretože som sa rozhol mobilnú aplikáciu zatiaľ nezverejňovať na Google Play pretože tam chcem implementovať vlastný server (bližšie info sa bude nachádzať vo výhľadoch do budúcnosti) aby sa dali otázky kategorizovať a druhý dôvod je zdroj dát. Ako zdroj dát som zvolil Google Places pretože ponúka čistú integráciu s Androidom, je od Googlu ale zároveň má neplatené konto denné limity a nechcel som aby ich aplikácia prekročila.

6.1 Výkonnostné a záťažové testy

Ako som spomenul na úvod, testovala sa prioritne analýza a persistencia bodov záujmu (ostatné časti aplikácie sa testovali behom vývoja priebežne). Kedže analýza bodov záujmu je závislá na pripojení k internetu tak výsledky testov nezávisia len na výkonnosti hardvéru zariadenia ale aj na kvalite pripojenia.
Testy prebiehali na následujúcich mobilných zariadeniach:

- LG Nexus 4 (2012)
- Sony Xperia Z5 (2015)

Výsledok testov analýzy bodov záujmu je uvedený v tabuľke 1 a práce s databázou v tabuľke 2.

Zariadenie	**LG Nexus 4**	**Sony Xperia Z5**
Analýza (Wi-Fi)	$\sim$ 46 sec.	$\sim$ 30,6 sec.
Analýza (LTE)	-	-
Analýza (HSPA+)	-	$\sim$ 33,6 sec.
Analýza (Edge)	-	$\sim$ 106,6 sec.

Tabulka 1: Výsledky testov analýzy bodov záujmu

Zariadenie	LG Nexus 4	Sony Xperia Z5
DB writing (batch)	$\sim$ 37,6 ms	$\sim$ 23,3 ms
DB writing	$\sim$ 80 ms	$\sim$ 34,6 ms
DB reading	$\sim$ 24,6 ms	$\sim$ 12,6 ms

Tabulka 2: Výsledky testov čitania z databázy

Z testov môžeme usúdiť, že pokiaľ má aplikácia relatívne dobré pripojenie k internetu (Wi-Fi, LTE, HSPA+) je svižná a výsledky sa líšia malým rozdielom. Pokiaľ je ale k dispozícii len pripojenie pomocou edge tak je aplikácia približne 3x pomalšia vzhľadom na dlhšie trvanie analýzy bodov záujmu kde výkon závisí na rýchlosti konektivity zariadenia.

Čo sa týka komunikácie s databázou tak môžeme taktiež povedať, že je aplikácia rýchla a dáta sú získavané z databázy behom niekoľko jednotiek až desiatok milisekúnd.
Pre zaujímavosť som uviedol porovnanie rýchlosti zapisovania do databázy za použitia transakcie kde je vidieť, že zapisovanie v rámci jednej transakcie trvá na zariadení LG viac ako 2x rýchlejšie ako bez použitia transakcie.
Chcel som tým poukázať, že kedykoľvek sa pracuje s lokálnou databázou na mobilnom zariadení a hlavne ešte z oveľa väčším množstvom dát s akým pracuje táto mobilná aplikácia je veľmi výhodne využiť transakciu aj v prípade, že to samotná aplikácia nevyžaduje s cieľom získať čo najlepší výkon.

Ešte by som rád spomenul, že v rámci zachovania svižného výkonu aplikácie s ohľadom na využitie procesora a pamäte aj na lacnejších mobilných zariadeniach sa vždy zobrazuje maximálne 100 bodov záujmu. Toto číslo mi prišlo po pár testoch za najvhodnejšie. Nie je to málo ale nie je to ani moc a aplikácia by mala fungovať svižne na väčšine mobilných zariadení.

6.2 Funkčnosť

Mobilná aplikácia podáva relatívne dobrú funkčnosť. Jediné obmedzenie je nutné spomenúť ohľadom dynamického generovania otázok. Tieto otázky sú generované na základe dát z wikipédie takže výsledok je závislý na samotnej dostupnosti wikipédie a prítomnosti dát o danom konkrétnom bode záujmu.
Takže sa môže stať, že aplikácia niekedy neúspešne vygeneruje otázky a užívateľovi podá informáciu, že sa žiadne vhodné otázky pre neho nenašli. Tento problém môže byť vyriešený implementáciou vlastného servera a odlišným prístupom ku generovaniu otázok. Tento návrh spomeniem vo výhľadoch do budúcnosti.

7 Záver

Cieľom práce bolo vytvoriť mobilnú aplikáciu, ktorá sa bude snažiť jednoduchým spôsobom obohatiť užívateľa o nové poznatky o rôznych zaujímavých miestach okolo neho prostredníctvom rôznych otázok a questov, ktoré môže zdieľať medzi svojmi priateľmi a tým ich napríklad motivovať aby sa pokúsili dokončiť podobný alebo rovnaký quest.
Na začiatku práce som sa venoval samotnému návrhu aplikácie, kde som popísal rôzne postupy, prístupy a knižnice, ktoré bude používať pri implementácii a podal popis tvorby užívateľského rozhrania použitého pre danú mobilnú aplikáciu a aj týmto spôsobom podal základné informácie o *material dizajne*, ktorým podľa môjho názoru urobil *Google* veľký pokrok a zatienil ním svojho konkurenta spoločnosť *Apple*.
V ďalšej časti textu som sa venoval implementácii mobilnej aplikácie kde som popísal a vysvetlil postupy a algoritmy, ktoré som použil a uviedol k niektorým príklady a náhľady prostredníctvom výpisov zdrojového kódu alebo diagramov.
Koniec textu bol venovaný testovaniu aplikácie a jej výkonnostným testom kde som následne získané výsledky zhodnotil a porovnal.

Aplikácia podáva relatívne dobré výsledky a užívateľovi dokáže podať zaujímavé poznatky o rôznych zaujímavých miestach aj keď dynamické generovanie otázok na základne vyhľadávacej frázy má svoje obmedzenia a jedným z nich je napríklad kategorizácia vedomostných otázok na základe obtiažnosti alebo detailnejšie prerozdeľovanie otázok len pre určitú záujmovú skupinu ľudí (napríklad žiaci prvého stupňa základnej školy).
Pre tieto spomenuté prípady sa javí byť najideálnejšie riešenie vlastný komplexný server, ktorý by sa o dané otázky staral a ktorý by sa neustále dopĺňal o nové a nové otázky pomocou kompletne nevrhnutého UI v spolupráci s ďalšími službami (napríklad aj s použitými Google Places).

Za zmienku ešte určite stojí použitie prepracovanejšieho algoritmu výberu bodov záujmu v nejakej novšej verzii aplikácie aby bol algoritmus menej závislý napríklad na použitej wikipédii a aby podával lepšie výsledky (napríklad použitie jedného z algoritmov, ktoré sú uvedené v zadaní tejto práce).

Toto by som videl ako *výhľady do budúcnosti* kde by som chcel spraviť z aplikácie startup, ktorý by bol v prípade úspechu použitý napríklad v školách.

Literatura

[1] Meier, Reto: Professional Android 4 Application Development

Wrox, 2012. ISBN-13: 978-1118102275

[2] Lee, Wei-Meng: Beginning Android 4

Wrox, 2012. ISBN 9781118199541

[3] Murphy, Mark L.: Beginning Android 2

Wrox, 2012. ISBN13: 9781430226291

[4] Aggarwal, C. C: Social Network Data Analytics

Springer, 2011. ISBN 978-1-4419-8462-3

[5] Brinton, Laurel J.: The Structure of Modern English: A Linguistic Introduction

John Benjamins Publishing Company, 2000. ISBN 978-90-272-2567-2

[6] The hyperonym problem revisited: Conceptual and lexical hierarchies in language generation [online]
Dostupné z <http://aclweb.org/anthology//W/W00/W00-1413.pdf>

[7] Places API Web Service [online]
Dostupné z <https://developers.google.com/places/web-service/search>

[8] Introducing JSON [online]
Dostupné z <http://www.json.org/>

[9] JSON: The Fat-Free Alternative to XML [online]
Dostupné z <http://www.json.org/xml.html>

[10] Material design [online]
Dostupné z <https://material.google.com/>

[11] About SQLite [online]
Dostupné z <https://www.sqlite.org/about.html>

[12] Calculate distance, bearing and more between Latitude/Longitude points [online]
Dostupné z <http://www.movable-type.co.uk/scripts/latlong.html>

[13] Using Longitude and Latitude to Determine Distance [online]
Dostupné z <http://mathforum.org/library/drmath/view/51711.html>

[14] Android 2.3 vs. Android 4.0 [online]
Dostupné z <http://www.androidauthority.com/android-2-3-gingerbread-vs-android-4-0-ice
>

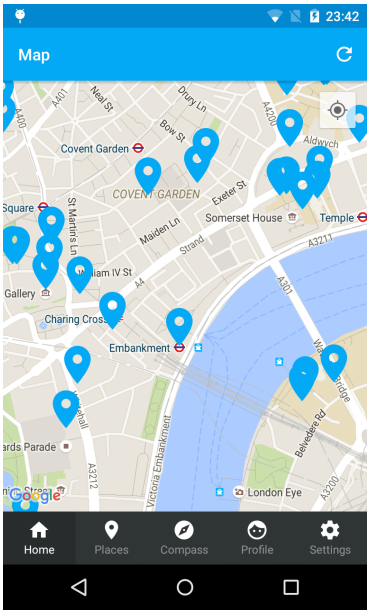
[15] TextBlob: Simplified Text Processing [online]
Dostupné z <http://textblob.readthedocs.io/en/dev//>

[16] What is WordNet? [online]
Dostupné z <https://wordnet.princeton.edu/>

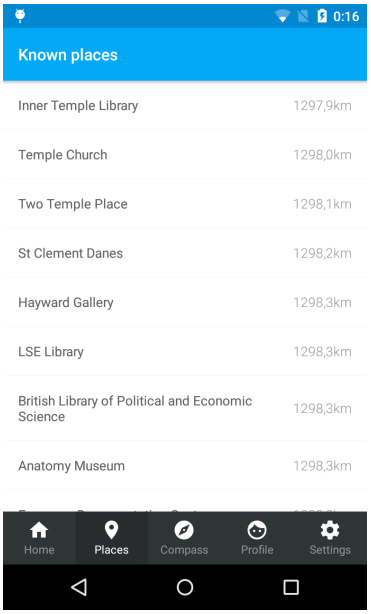
[17] Introduction to WordNet: An On-line Lexical Database [online]
Dostupné z <http://wordnetcode.princeton.edu/5papers.pdf/>

A Kompletný návrh užívateľského rozhrania

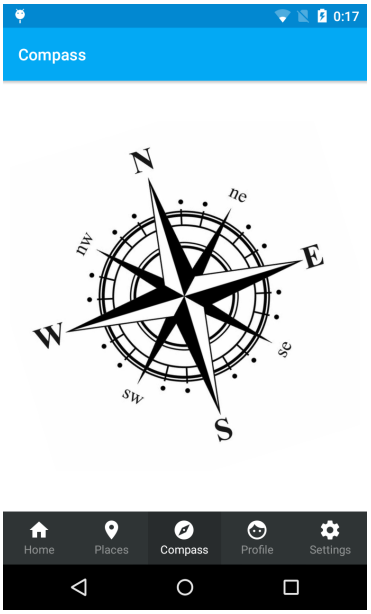
Na obrázkoch nižšie je ukážka kompletného užívateľského rozhrania mobilnej aplikácie. Cieľom bolo vytvoriť jednoduché UI s dobrým UX tak aby bola aplikácia intuitívna a užívateľ sa v nej hneď nestratil.



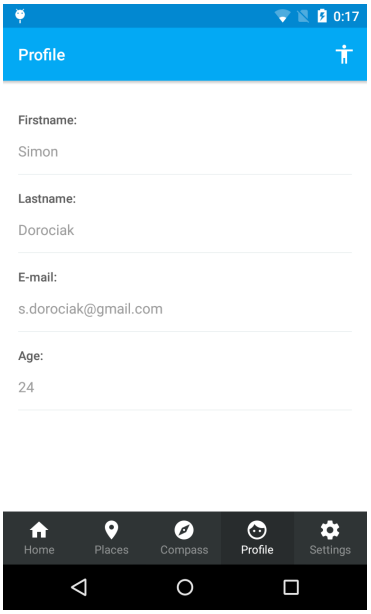
Obrázek 23: UI: Obrazovka 'Home'



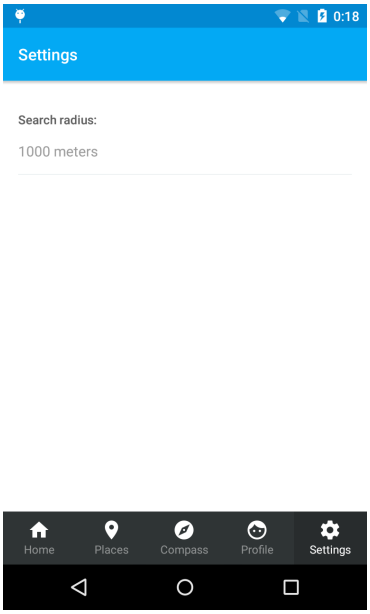
Obrázek 24: UI: Obrazovka Miest Záujmu



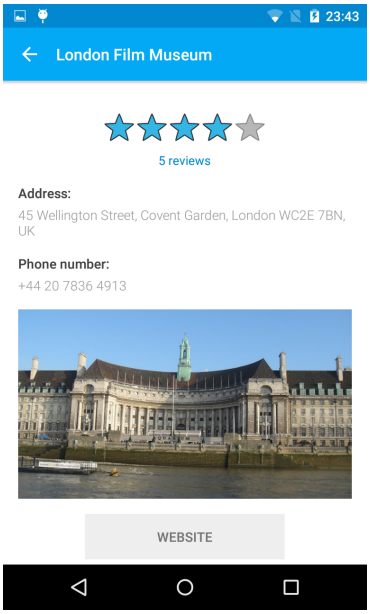
Obrázek 25: UI: Obrazovka Kompasu



Obrázek 26: UI: Obrazovka Profilu

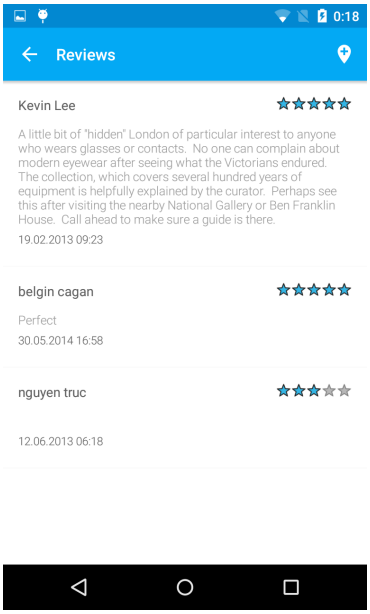


Obrázek 27: UI: Obrazovka Nadstavení

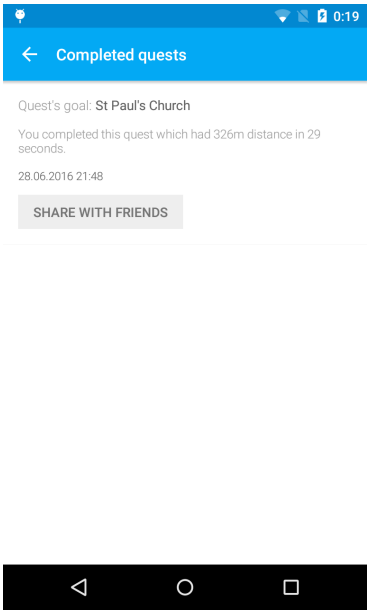


Obrázek 28: UI: Obrazovka Detailu POI

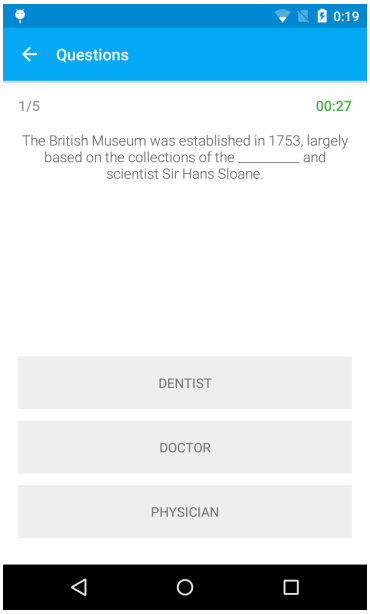
Obrázek 29: UI: Obrazovka Wiki Detailu



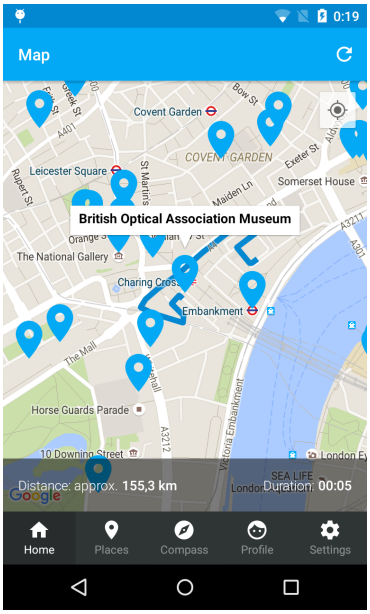
Obrázek 30: UI: Obrazovka Recenzií POI



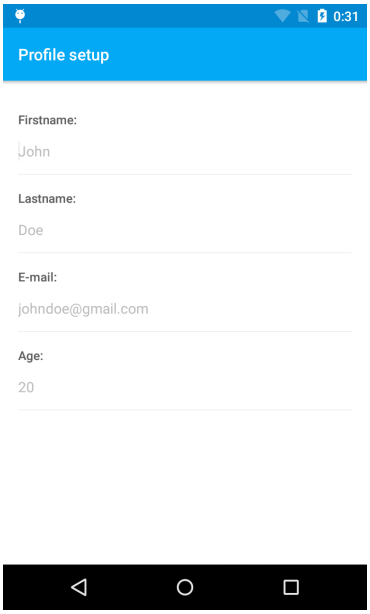
Obrázek 31: UI: Obrazovka Questov



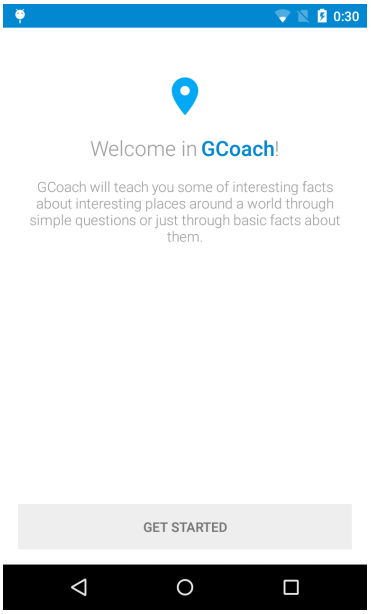
Obrázek 32: UI: Obrazovka Detailu Otázky



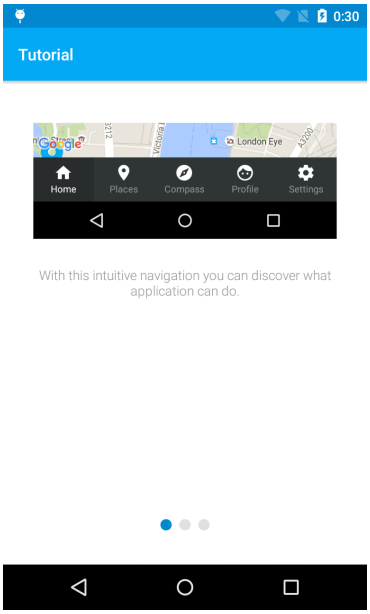
Obrázek 33: UI: Obrazovka Questu



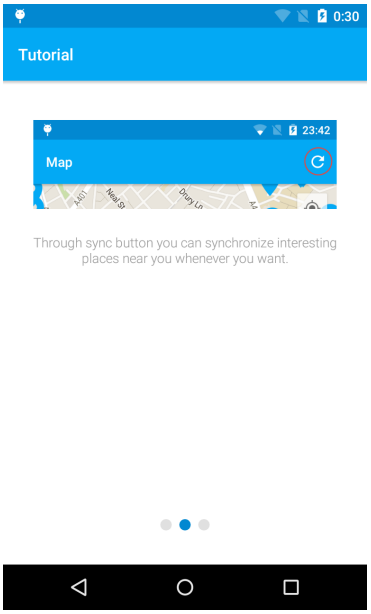
Obrázek 34: UI: Obrazovka Profilu 2



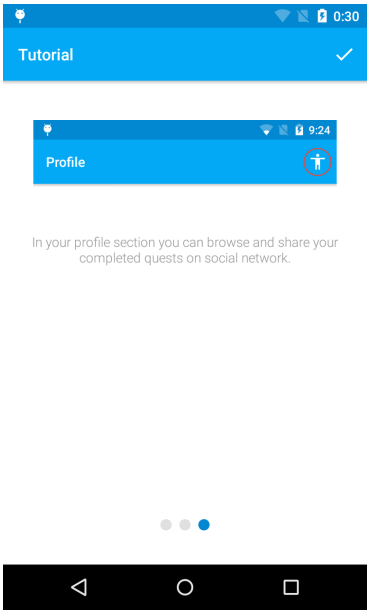
Obrázek 35: UI: Obrazovka 'Welcome'



Obrázek 36: UI: Obrazovka Tútoriálu 1



Obrázek 37: UI: Obrazovka Tútoriálu 2



Obrázek 38: UI: Obrazovka Tútoriálu 3